NEGOTIATION-BASED DECENTRALIZED CONFLICT RESOLUTION IN MULTI-AGENT PATH FINDING

A Thesis

by

Cihan Eran

Submitted to the
Graduate School of Sciences and Engineering
In Partial Fulfillment of the Requirements for
the Degree of

Master of Science

in the
Department of Computer Science

Özyeğin University
December 2021

NEGOTIATION-BASED DECENTRALIZED CONFLICT RESOLUTION IN MULTI-AGENT PATH FINDING

Approved by:

Assistant Professor Reyhan Aydoğan,
Advisor
Department of Computer Science
Özyeğin University

Associate Professor Ömer Korçak
Department of Computer Software
Marmara University

Assistant Professor M. Furkan Kıraç
Department of Computer Science
Özyeğin University

Date Approved: 29 December 2021

To my family and friends

ABSTRACT

This thesis addresses the problem of Multi-Agent Path Finding problem where multiple agents aim to reach their destination in a grid world without any collision. It aims to provide a solution achieving good trade-off between the privacy of the agents and the effectiveness of solutions. Accordingly, a token-based bilateral negotiation approach is presented to solve this problem in a distributed way. The proposed approach is evaluated empirically in various scenarios by comparing it with state-of-the-art centralized approaches such as Conflict Based Search and its variants. The experimental results showed that the proposed approach can find conflict-free path solutions albeit suboptimally, especially when the search space is large and high-density, whereas centralized approaches struggle to find optimal solutions. Despite being outperformed by suboptimal centralized solvers, the proposed decentralized approach can achieve considerable results with naive agents by sharing minimal information about themselves. The proposed approach also enables agents to have their autonomy; thus, the proposed approach is convenient for MAPF problems involving self-interested agents.

ÖZETÇE

Bu tez, çoklu etmenli sistemlerde yol planlaması problemine, müzakere yöntemi tabanlı bir yaklaşım geliştirmeyi hedefliyor. Bu yaklaşımda, sistemdeki etkenlerin verilerinin gizliliği ve çözümlerin efektifliği arasında kayda değer bir denge kurulması amaçlanmıştır. Bunun için, jeton tabanlı ikili müzakere protokolü ve bu protokol ile uyumlu müzakere stratejileri sunulmaktadır. Önerilen yaklaşım, çeşitli senaryolarda, Çakışma Tabanlı Arama (CBS) ve benzer gelişmiş merkezi sonuç üretme çözümlerine karşı sonuçları değerlendirilmiştir. Deney sonuçlarında sunulan dağıtık sorun çözme yaklaşımının merkezil çözüm yaklaşımlarına karşın kayda değer sonuçlar üretebildiğini göstermektedir. Önerilen yaklaşım, sistem etmenlerinin kendi karar vermelerini sağlamaktadır. Bu sebepten ötürü, bu yaklaşım, bireyselliğini gerektiren durumlar için idealdir.

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude and respect to my advisor, Assist. Prof. Dr. Reyhan Aydogan for her guidance, support, and patience.

I would also like to thank my thesis committee members Assoc. Prof. Dr. Ömer Korçak and Assist. Prof. Dr. M. Furkan Kıraç for their invaluable feedback and valuable time.

I would like to extend my deepest appreciation and gratitude to Onur Keskin and Furkan Cantürk for their tireless efforts and the sleepless nights they have spent in order to support me on the analysis of results. Additionally, I would also like to thank all members of Interactive Intelligent Systems especially Berk Buzcu, Mert Özgün, Umut Çakan and Furkan Arslan.

Finally, I am grateful to my family for their unending, unconditional support through out my journey, and enduring along with me. I would not be able to begin to imagine concluding my study without their fervent support.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

π_i Sequence of vertices for A_i

A Agent

EC Estimated cost of a given path, P_c

g_i Goal vertex for A_i

H_T^j Heat Map for Agent j at a given time T

N_c Neighbor agents excluding the opponent in the field of view

$O_{opponent}$ Opponent offer

$P_{current}$ Current path in FoV

P_c A given path whose cost will be estimated

$P_{remaining}$ Agent's remaining path to destination

P_{Space} Sorted path space

s_i Starting vertex for A_i

$T_{remaining}$ Agent's remaining tokens count

CHAPTER I

INTRODUCTION

Technological advancements in the last decades have enabled autonomous robots and vehicles to carry out various tasks such as surveillance and transportation. They may need to navigate from one location to another to achieve their goal. Imagine an environment in which hundreds of autonomous robots aim to reach certain locations. Such an environment requires a coordination mechanism to avoid some potential collisions. This problem, allocating conflict-free paths to agents so as to navigate safely in an environment, is well-addressed in the field of Multi-Agent Systems and known as Multi-Agent Path Finding (MAPF) problem [1]. A vast number of studies tackle this problem; some propose a centralized solution while others focus on decentralized solutions [2].

Centralized solutions rely on full access to all relevant information regarding the agents and properties of the given environment so that a global solution can be derived. In contrast, decentralized solutions decouple the problem into local chunks and address the conflicts locally [3]. Without any time constraints, centralized approaches can find optimal solutions if there exist. However, the performance of centralized solution approaches can suffer in high density, and complex environments [4, 5]. Besides, complete information may not always be available due to the limitation of communication, sensors, or privacy issues. On the one hand, decentralized approaches can deal with uncertainty and scalability issues and produce admissible solutions. However, they may overlook a potential optimal solution and end up with suboptimal solutions. Accordingly, this thesis aims to resolve the conflicts occurring in multi-agent path-finding problems and introduces a negotiation-based decentralized conflict resolution approach.

1.1 Motivation

Multi-agent systems consist of intelligent agents that observe and interact with the environment they are in [6]. An agent is an entity that resides in the environment and interacts with the environment through its actuators and observes the environment with its sensors. These agents can be of any form, so long as they interact with the environment and behave autonomously. Each agent has its own goals to achieve, and to reach their goals, agents can often interact with both environment and other agents in the environment in a cooperative and/or competitive manner. Agents can engage in negotiations, voting, or auctions to reach an agreement in their communications.

This work pursues a decentralized approach to the MAPF problem targeting a good trade-off between privacy and effectiveness of the solutions. As agents can resolve their conflicts for varying problems from resource allocation to planning through negotiation [7, 8, 9, 10], we advocate to solve the aforementioned problem in terms of negotiation and accordingly propose a token-based alternating offers protocol.

1.2 Contributions

In the proposed approach, agents share their partial path information with only relevant agents close to them to some extent. If they detect any conflict on their partial path, they encounter a bilateral negotiation to allocate required locations for certain time steps. This study introduces a variant of alternating offers protocol enriched with token exchanges to govern this negotiation. By enforcing the tokens' usage, the protocol leads agents to act collaboratively and search unexplored paths so that there is no conflict anymore. Two negotiation strategies are presented in line with the protocol, namely Path-Aware and Heat Map strategies. While the Path-Aware negotiation strategy focuses on the agent's information on their own paths and makes bids considering the length of its final path, the Heat Map strategy utilizes other agents' available path information in its field of view to make bids avoiding potential future conflicts.

There are a few attempts to solve the MAPF problem in terms of negotiation [11, 12, 13]. Either they require sharing full path information with others, or they consider one-step decisions such as who will move to a certain direction at the time of conflict, or they aim to resolve the conflict in one shot (i.e., collaborate or reject). A detailed analysis of the proposed approach is also made in the following sections. In contrast, our approach aims to reduce the complexity of the problem by resolving the conflicts in subpath plans iteratively instead of the entire path plans, thereby respecting the agents' privacy to some extent.

1.3 Thesis Outline

The thesis is organized as follows, Chapter 2 gives information about the previous works in the literature that address the concept of decentralized solutions and utilizing negotiation as conflict resolution for the multi-agent path finding problem. Chapter 3 describes the framework, negotiation protocol, and the agents that are presented in this thesis. Chapter 4 discuss the experimental results showing how well our agent performs in contrast to the centralized solutions on different scenarios and environment configurations. Finally, Chapter 5 concludes this thesis by briefly discussing the results and future direction of the research.

CHAPTER II

RELATED WORK

Conflict resolution approaches for MAPF problems can be classified according to the centralization of solution mechanisms and cooperation of agents. Centralized solution approaches to path-finding of cooperative agents provide optimal plans [14]. If a trusted center with the information of all agents moving in a known environment and the ability to operate all of them are not available, negotiation can be used for a conflict resolution mechanism [15, 16, 13, 12, 17]. One negotiation approach to allocating resources to multiple parties is Combinatorial Auction (CA). To resolve conflicts between self-interested agents in an environment, Amir *et al.* reduce MAPF problem to CA and implements IBundle, an iterative CA algorithm [18], for MAPF [15]. One crucial factor for the effectiveness of CA algorithms is the trustworthiness of self-interested agents about their utilization. Considering this issue, Amir *et al.* propose Vickrey-Clarke-Groves (VCG) auction for MAPF, a strategy-proof auction mechanism for manipulation attempts by the agents. In this IBundle auction, the auctioneer is exposed to a computational burden as agents submit their all-desirable bundles, which requires even impractical auction time. Addressing this limitation of IBundle, the auction mechanism by Gautier *et al.* can provide a feasible allocation more likely by allowing agents to submit a limited number of bundles and so terminates in fewer time [16]. Our solution compared to these CA-based conflict resolution for MAPF is free of trustworthiness and computational load concerns as it applies bilateral negotiation to resolve each conflict.

Key challenges of addressing the MAPF problem within the decentralized method can be summarized as establishing a solution framework for agents to use while interacting with the environment, defining an interaction protocol between agents, and designing agents that can reach a solution. Purwin *et al.* propose a decentralized framework where cooperative agents allocate portions of the environment in which they can move. Using these frameworks, agents send information to others via a network and keep information about all. Here, agents who collide update their trajectories with a pairwise agreement based on a priority rule. Similarly, agents broadcast information in our framework while seeking a conflict-free path, resolving conflicts bilaterally. However, the revealed information is local, and it is only achievable by agents in the broadcast range.

The work of Pritchett *et al.* defines a bilateral negotiation structure to prevent collisions in air traffic [12]. Each round of the negotiation session offers cost increases until an agreement is reached. This offering mechanism forces agents to concede over time. In contrast, our Token-based Alternating Offers Protocol (TAOP) does not expose a catalyst to reduce the number of offers, and it is left open to be addressed by agent strategy designs. Different from bilateral conflict resolution approaches, Sujit *et al.* focus on solving a task allocation problem in their work by using a multilateral negotiation [17]. Besides, in their work, agents utilize the presented token structure to determine whose offer to be accepted in a deadlock situation that might happen. The agent with the least number of tokens is supposed to accept the other side's offer. Whereas in our framework, token usage is dedicated to penalizing repeated offers.

Inotsume *et al.* demonstrate a bilateral negotiation-based approach to MAPF [11]. In that approach, agents submit their desired paths to an area manager before moving their initial positions. The area manager is responsible for keeping control of a reservation table and responses path requests by agents based on this table to prevent any intersecting paths. The requests are accepted on a first-come, first-serve basis. Using such a path reservation system deviates from the proposed decentralized approach in the paper. Agents can request already reserved areas from corresponding agents by offering some tokens. In this framework, agents have the objective of maximizing tokens alongside path cost minimization, whereas our solution uses tokens as an instrument to manage the negotiation behaviour of any agent design to reach an agreement.

MAPF Problem requires all of the agents to reach their destination. Therefore, they have to behave in a way that complies with the social good to some extent because allowing agents to behave selfishly usually leads to a Nash Equilibrium [19]. To motivate self-interested agents in a network to behave in a socially desirable way, taxation is a classical approach to diminish congestion in the network [20]. Considering MAPF Problem for self-interested agents, Bnaya *et al.* [21] propose Iterative Taxing Framework to detect where, when, and how much additional cost is imposed to agents to change their paths so that they are discouraged from using paths that would collide with others. Their results show that a substantial total cost decrease is obtained compared to a solution scheme depending on a selfish replanning procedure. However, they do not benchmark with optimal solution cost. Thus, the efficiency of the proposed framework needs to be investigated further. Following the system efficiency, Ramos *et al.* introduce a toll-based multi-agent reinforcement learning approach to minimize total congestion and can achieve system-efficient results. In this study, tolls are used to divert drivers from congested routes. Similar to these two studies, our proposed approach for MAPF aims to urge agents to prefer alternative optimal paths or individual suboptimal paths when they face a density of other agents in their next subpaths requiring more tokens to resolve conflicts there.

Besides providing a scalable multi-agent planning framework for path-finding problems among self-interested agents, we also address the privacy aspect in the multi-agent planning domain with our study. In the privacy-concerned planning concept, agents have the task of searching and are enforced to preserve disclosure of sensitive data while resolving conflicts of interest. MA-STRIPS [22] model provides a privacy-preserving planning scheme to control what part of the information is to be shared with all agents. However, this planning paradigm reveals information among all agents instead of transmitting data only between the agents subject to a conflict of interest. Bonisoli *et al.* provide a generalized form of MA-STRIPS by establishing pairwise privacy among agents [23]. In their setting, agents can specify a subset of agents intended to share information. Their approach is suitable for multi-agent problems, including individual interest protection. Similarly, in our problem, agents need to collaborate with other agents in case of future conflicts, although they are all self-interested. Therefore, path-seeker agents have to reveal local information.

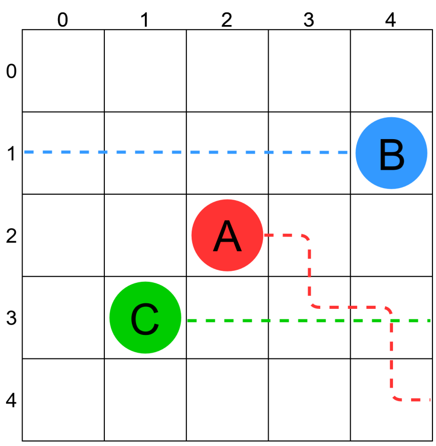
Our decentralized MAPF framework follows a geographical manner to condition what part of the information is shared with which agents. Such local information revealed with some agents is needed in any case because the addressed problem has a concern of no collision, and it requires proactive responses (i.e. replanning) by the stakeholder agents before a reasonable time.

CHAPTER III

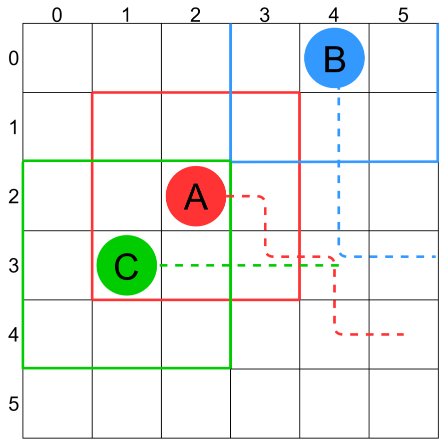
PROPOSED FRAMEWORK

Multi-Agent Pathfinding (MAPF) as defined in [1], is the problem of assigning conflict free paths to agents from their respective starting locations to their destinations. Formally, we have k agents denoted by $A = \{ A_1, A_2, ..., A_k \}$ navigating in an undirected graph $G = (V, E)$ where starting and destination location for each A_i are denoted by $s_i \in V$ and $g_i \in V$ respectively. The path of each agent A_i is denoted by π_i, a sequence of vertices indexed by each time step $0 \to n$, $(s_i^0, ..., g_i^n)$. $\pi_i^t \in V$ corresponds to the location of A_i at time step t. At any time step, the agents cannot be located in the same vertex (i.e., $\pi_i^t \neq \pi_j^t \; \forall \; i \neq j$) and traverse the same edge. A swapping conflict occurs when agents in adjacent cells want to move towards each others' location (i.e., $\pi_i^t = \pi_j^{t+1} \wedge \pi_i^{t+1} = \pi_j^t \; \forall \; i \neq j$).

In this thesis, agents are located in a $M \times M$ grid-like environment where each cell corresponds to a vertex as illustrated in Figure 1. Each agent has an initial path to reach their destination, shown by dashed lines in the grid. For example, we have three agents A, B, and C whose planned paths are colored in their respective colors (i.e., red, blue, and green) in Figure 1a. Here, an agent in a cell can only move to their vertical and horizontal adjacent cells (i.e., cardinal directions). For instance, Agent A located in $(2, 2)$ can move to one of the following cells: $(1, 2)$, $(3, 2)$, $(2, 1)$, and $(2, 3)$. The example in Figure 1a shows a conflict between Agent A and C at time step t=2 (cell $(3, 3)$). They need to resolve this conflict to achieve their goals. Final solutions will be evaluated under the objective of some of the individual costs, $\sum_{i=1}^{k} |\pi_i|$ where individual cost of each agent i corresponds to their path length denoted by $|\pi_i|$.



(a)



(b)

Figure 1: Example Environment & Field of View Representation

There are some variants of this problem. In *disappear at target* MAPF problems, agents can disappear from the environment when they reach their destinations while agents stay at the destination in *stay at target* MAPF problems (i.e., $\pi_i^t \neq \pi_i^{t+1}$ whereas $\pi_i^t \neq g_i$.). In another variant, agents cannot wait at a given time step while they can either wait or move in another variant. Accordingly, this thesis studies four different variants of this MAPF problem as follows:

- Case-1: Agents cannot wait at any time till reaching their destination. They stay at target and act as obstacles to other agents.
- Case-2: Agents can wait at any time till reaching their destination and stay at the target as an obstacle for other agents.
- Case-3: Agents cannot wait at any time till reaching their destination, and they disappear at target.
- Case-4: Agents can wait at any time till reaching their destination and disappear at target.

For the above problems, we present a decentralized solution in which agents autonomously negotiate with each other in order to refine their paths to avoid possible collisions. The main challenge is to deal with the uncertainty about the environment due to the limited capacity of the sensors, communication, or some privacy concerns. Most real-world applications are partially observable where the agents can perceive some relevant aspects of the environment. For instance, a robot may not perceive all objects far from its current location. Light detection and range finding sensors can detect up to a certain distance. Similarly, wireless communication systems also have limited communication capabilities. Agents may not exchange information with each other if their distance is above a threshold value. Besides, complete information is not always available due to the characteristics of the environment. For example, drivers in traffic do not know where other drivers are going. Furthermore, agents may be reluctant to share all information due to their privacy. For instance, they may not be willing to reveal their destination.

3.1 Framework

In the proposed framework, agents are located in a grid as shown in Figure 1b. Initially, each agent knows only their starting location, destination, and a path plan to reach their destination. Those planned paths are shown in colored dot lines in the grid for each agent. For simulating the aforementioned partially observable environment, we adopt the concept of *field of view*. The framework enables agents to access a limited portion of other agents' planned paths within a certain proximity and share their own. In other words, an agent's field of view determines the scope of its communication and perception capacity. An agent can only observe and communicate with other agents within its field of view. That is described as a certain number of cells d from its location. For instance, when d is equal to 1, a red rectangle for Agent A shows the field of view boundary. In such a case, Agent A can receive/send information from/to only agent C located in the scope of Agent A's field of view and vice versa. However, in the given snapshot, agent B cannot communicate or see other agents at that time.

In this framework, agents broadcast their sub-planned path. Agents are free to determine to what extent the path is to be shared with other agents in the field of view. In our experiments, agents share their current subpath with a length of $2d$. If any conflict is detected by one of the agents, they can engage in a negotiation session. For example, when d is equal to 1, agents will share their current subpath with a length of 2 (i.e., its next two moves) with the agents located the scope of their field of view. Agent A broadcasts its current subpath as $Broadcast : [\pi_A^{t=1} = (3, 2), \pi_A^{t=2} = (3, 3)]$ while Agent C shared its own as $Broadcast : [\pi_C^{t=1} = (2, 3), \pi_C^{t=2} = (3, 3)]$. Since agents would detect a conflict in the vertex $(3, 3)$ at t=2, Agent A and C start negotiating the allocation of vertices on their path since they detect a conflict in $(3, 3)$.

When an agent detects a conflict with more than one agent, which negotiation to be held first is determined in *first come first serve* basis. For example, if d is 2, Agent B and Agent C will share their subpaths with a length of 4 with Agent A. Agent A may first negotiate with Agent B if Agent B's message has been received before Agent C's one. Afterward, it can encounter a bilateral negotiation with Agent C. After carrying out any successful negotiation, agents will update their path accordingly. Several negotiation sessions might be held until resolving current conflicts. If there are no conflicts in the current field of view, agents move to the following location in their path. Once an agent reaches its desired destination, it will not encounter a negotiation anymore. The negotiation between agents is carried out according to the proposed token-based negotiation protocol. The details of this protocol and a specific bidding strategy mainly designed for this protocol to tackle the MAPF problem are explained in the following sections.

3.2 Token-based Alternating Offers Protocol

The proposed framework requires agents to negotiate to resolve conflicts in their paths. The conflict may occur between two or multiple agents at a given time. When it happens among more than two agents, we can formulate it as multiple bilateral negotiations and consider it a multilateral negotiation. As it may be harder to find a joint agreement, especially when the number of participants is high [24], the proposed approach aims to solve the conflicts in multiple consecutive bilateral negotiations. For simplicity, agents perform their bilateral negotiations consecutively. That is, a new negotiation can start after completing the previous one.

In the proposed approach, when there is a conflict in two agents' sub-path, agents negotiate to allocate the relevant vertices for certain time steps. Following the previous example illustrated in Figure 1b, Agent A may claim to allocate the vertices at $(3, 2)$ at time $t = 1$, $(3, 3)$ at time $t = 2$ while Agent C may aim to allocate the vertices at $(2, 3)$ at time $t = 1$, $(3, 3)$ at time $t = 2$. Depending on how the negotiation proceeds, they may concede over time and change their request on vertex allocations to come up with an agreement. If agents find an agreement, they are supposed to obey the allocation for the other party. Agents are free to change their path as long as their current path allocation does not violate the agreed vertex allocation for the other party. For example, when Agent C accepts Agent A's vertex allocation for time steps $t = 1$ and $t = 2$, Agent C confirms that it will not occupy those vertices allocated by Agent A for the agreed time steps.

The interaction between agents needs to be governed by a negotiation protocol. In automated negotiation, agents mostly follow the Stacked Alternating Offers Protocol [25] in which they exchange offers in a turn-taking fashion until reaching a predefined deadline. This protocol does not force the agents to come up with an agreement. If both agents are selfish, they may fail the negotiation. However, finding a consensus plays a crucial role in MAPF. Therefore, agents preferably follow a protocol leading them to reach an agreement. Accordingly, we introduce a novel token-based negotiation protocol namely *Token-based Alternating Offers Protocol* (TAOP) inspired from Monotonic Concession Protocol (MCP) [26] and Unmediated Single Text Protocol (USTP) [27].

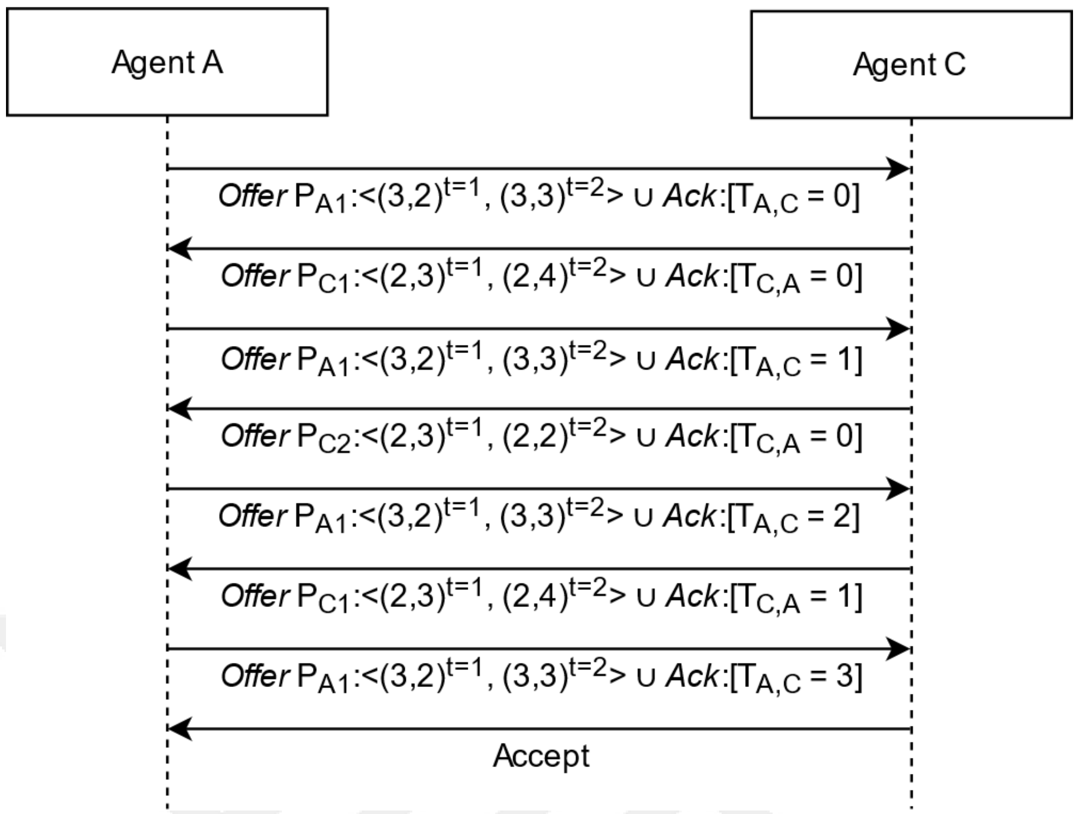


Figure 2: Example Interaction between Negotiating Agents

According to Monotonic Concession Protocol, agents make simultaneous offers so that either they can stick to their previous offer or make a concession. If both parties stick to their previous offers, the negotiation ends without consensus. Otherwise, agents continue negotiation until reaching an agreement or fail the negotiation. This protocol leads agents to complete the negotiation without setting a predefined deadline. However, there is a high risk of ending up with a failure. In Unmediated Single Text Protocol, agents interchangeably become a proposer or voters during the negotiation. Initially, some tokens are given to each agent, where agents can use those tokens to override others' reject votes. One agent starts with a random offer, and the other agent votes to accept or reject it. It is considered the most recently accepted bid if the other agent accepts. This interaction is repeated multiple times, and the most recently accepted bid is updated over time. At the end of the negotiation, the most recently accepted bid is considered the agreement. Here, the tokens incentivize truthful voting of agents to not manipulate the system by rejecting all offers. Since this protocol is particularly designed for large-scaled negotiation problems, the generated bids are variants of an initial random offer, not directly applicable to our problem. On the other hand, the token idea can enforce the agents to concede over time in a fairway.

The proposed token-based alternating offers protocol is a variant of alternating offers protocol enriched with token exchanges. One of the agents initiates the negotiation with an offer. The receiving party can accept this offer, make a counteroffer, or end the negotiation without agreement. The main difference is that agents cannot repeat their previous offers unless they pay for them. The protocol assumes that each agent owns a predefined number of tokens, $\mathcal{T}$. Those tokens enable an agent to make one of its previous offers during that negotiation. Unlike Monotonic Concession Protocol, agents are not required to make conceding moves. The essential requirement for agents is to make unproposed offers during the negotiation or pay tokens to repeat an offer. In addition to the given offer, agents send an acknowledgment message specifying the number of tokens to be used to repeat an offer previously made by the same agent. The general flow of the proposed protocol is given below:

1. One of the agents makes an offer specifying its request to allocate some vertices for specific time steps and sends an acknowledgment message regarding the usage of its token in the current negotiation. Initially, the usage of tokens is set to zero.

2. The receiving agent can take one of the following actions:

   - ends the negotiation without any consensus.
   - accept the received offer and complete the negotiation successfully.
   - makes an offer specifying the vertices allocation for itself that has not been offered by that agent yet and sends the acknowledgment denoting the accumulated usage of its tokens.
   - can repeat one of its previous offers, increase the usage of its tokens by one, and send the token acknowledgment message.

3. If the agent accepts or ends the negotiation, negotiation is finished. The accepting agent receives tokens amounting to the calculated token usage difference from its opponent, $min(\mathcal{T}_{opp} - \mathcal{T}_{self}, 0)$ where $\mathcal{T}_{opp}$ and $\mathcal{T}_{self}$ denote the total number of tokens used by the opponent and the accepting agent during the entire negotiation respectively. If the accepting agents spend more tokens than their opponent, it does not receive any tokens. Otherwise, the receiving agent can take any action mentioned in Step 2.

Considering the scenario given in Figure 1b, an example negotiation trace between Agent A and C is illustrated in Figure 2. Agent A initiates the negotiation with its offer P_{A1} requesting to claim the vertices $(3, 2)$ for $t = 1$ ad $(3, 3)$ for $t = 2$. Agent C does not accept this offer and makes its own offer specifying its allocation, such as $(2, 3)$ and $(2, 4)$. Since Agent A insists on its previous offer, it increases its token usage by one. As seen from the example, agents send an acknowledgment message and offer each turn. Agent C accepts Agent A's offer in the fourth round. It confirms that Agent C will not move to $(3, 2)^{t=1}$ and $(3, 3)^{t=2}$. In return, Agent A will pay 2 tokens (T_A-T_C). It is worth noting that the token exchange is performed at the end of the negotiation, depending on who accepts the offer. If an agent needs to pay tokens but has insufficient tokens, the agreement is not committed (i.e., negotiation fails).

3.3 Negotiation Strategies

We introduce three bidding strategies, namely *Random*, *Heat Map* and *Path-Aware* to evaluate the performance of the framework. Note that the agents are called with the name of the bidding strategy that they adopt. Random Agent is considered as a baseline agent for our evaluation. While Path-Aware Agent takes its available path and token information into account, Heat Map Agent aims to reduce the potential conflicts and prefers the low-density paths, as explained below.

3.3.1 Path-Aware Negotiation Strategy

Existing negotiation strategies focus on only which offer to make at a given time and when to accept a given offer [28]. Therefore, there is a need to design a new strategy taking token exchanges into account. Hereby, we propose a negotiation strategy determining when to repeat an offer or generate a new offer. The proposed strategy, namely *Path-Aware* negotiation strategy, aims to utilize the information available to determine when to insist on its current path. It is worth noting that each agent generates its possible paths leading them to their destination by using the A-Star Algorithm by taking the non-conflicting paths of the other agents in its field of view. That is, if there are three agents in the field view and our agent has only conflict with one of them, it generates all possible paths that would not cause an additional conflict with the other agent where the agent does not have any conflict initially. Those possible paths constitute the agent's bid space. Afterward, they sort those paths in descending order concerning their path cost.

Algorithm 1 describes how an agent negotiates according to Path-Aware Negotiation Strategy. At the beginning of the negotiation, the current path in the field of view ($P_{current}$) is the relevant part of the optimal path (i.e., the shortest path to its destination). It corresponds to the first offer in the negotiation. When the agent receives an offer from its opponents, it checks whether it is possible to generate a path of equal length or shorter than its current path to the destination (Line 1). If so, it accepts its opponent's offer (Line 2). Note that the path generation function takes the opponent's offer, $O_{opponent}$, as a constraint while generating its best possible path to the destination. If the best possible path in line with the opponent's bid is better than or equal to the current path, then the agent accepts the opponent's bid. If the agent's remaining tokens are greater than the length of the remaining path to the destination (Line 4), it decides to repeat its previous offer and updates its remaining tokens accordingly (Line 5). Recall that for each repetition, the agent needs to use one token. Otherwise, it concedes and sets the following possible best path from the sorted path space P_{Space}, as its current path in its field of view (Line 7). Accordingly, the agent offers its previous path in the field of view $P_{current}$ (Line 9). Note that agents must concede if they do not have any tokens left.

Algorithm 1: Negotiation Strategy of Path-Aware Agent

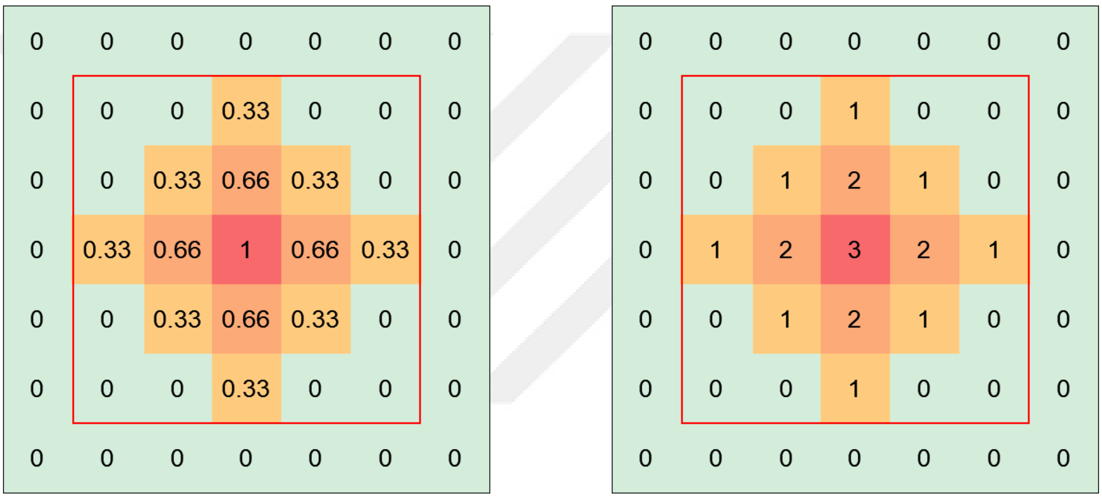
Data:
$T_{remaining}$: Agent's remaining tokens count
$P_{remaining}$: Agent's remaining path to destination
$P_{current}$: Current path in FoV
P_{Space}: Sorted path space
$O_{opponent}$: Opponent offer

1 **if** $|P_{current}| \geq |generatePath(O_{opponent})|$ **then**
2   accept()
3 **else**
4   **if** $T_{remaining} > |P_{remaining}|$ **then**
5         $T_{remaining} - -$
6   **else**
7         $P_{current} \leftarrow P_{Space}$.next()
8   **end**
9   offer($P_{current}$)
10 **end**

3.3.2 Heat Map Negotiation Strategy

The Path-Aware strategy focuses more on the opponent's bids and own remaining path and token information. Therefore, the density of the fields in the environment is neglected during the bidding process. Thus, the agreed paths may lead to the following conflicts. The Heap Map bidding strategy incorporates this kind of environmental information into evaluating bids. This strategy uses the available field of view data to generate a heat map of the terrain around the agent and detects high-density areas. Consequently, it avoids generating a path on the high-density area as the cars avoid the potential traffic jams by preferring the low-density routes in the traffic. In other words, the agent aims to foresee which part of the map would be the best move leading to less collision by analyzing the received path information from other agents. The generated heat map is built based on the broadcasted information of other agents within the field of view. Based on the given path information, it generates a heat map for each time step by considering only the paths of the agents other than the current opponent in the field of the view. It considers each heat map while calculating the estimated cost of any path.

Figure 3 shows an example of a heat map created for the agent located in the middle of the grid. When the location information of an active agent is detected by other agents through their broadcasts, during the heat map generation, the detected agent at time t generates a heat signature from their location, as seen in the figure. Here, the values are normalized concerning the field of view.



(a) Normalized Heat Values for Agent (b) Real Heat Values for Agent

Figure 3: Heat Generated by an Agent

To get a better understanding of how the agent uses a heat map, we illustrate a toy example depicted in Figure 4. According to our example, Agent A and Agent B start negotiation because they detect a swap conflict on their paths. Therefore, when it estimates the cost of each possible path, it considers only the heat map of Agent C. As seen from Figure 4, Agent A generates the heat map of Agent C for t=1 and t=2 based on the broadcasted path information by Agent C. Let us assume it will calculate the estimated cost of its initial path, $(2,2)$, $(2,3)$, $(3,3)$. At t=1 it is is supposed to be at $(2,3)$ and the value of this cell is 0.33, while at t=2 it will be $(3,3)$ and its heat map value is also 0.33 according to the heat map of the Agent C. The path cost would be estimated as 0.66 (=0.33+0.33).

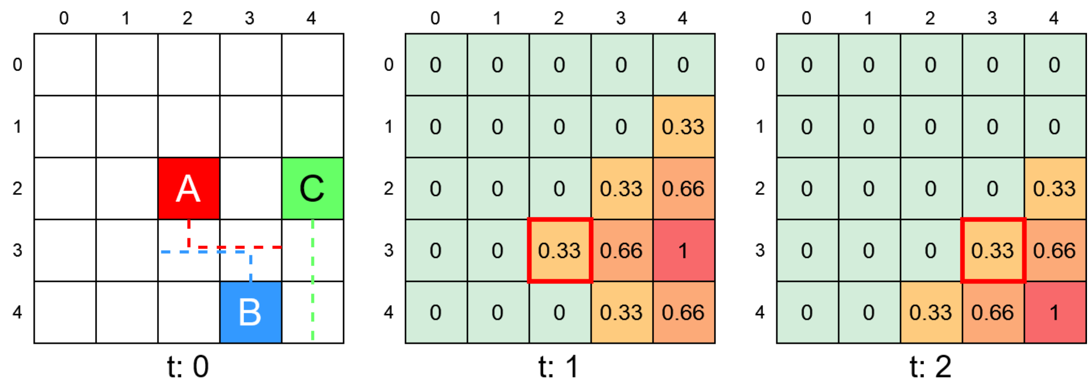


Figure 4: Heat Map of Agent C According to Agent A

When multiple agents are present in the heat map, the values of the cells are added up, as can be seen in Figure 5 depicting a portion of the heat map generated by Agent A at time step t for its negotiation session with Agent D. It is assumed that Agent A can listen to broadcasts of Agent B, C, and D. Since Agent A is engaged in a negotiation session with Agent D, the heat generated by Agent D is ignored. However, the paths of Agent B and C are assumed to be steady for the duration of the negotiation. Therefore, their broadcasted information is used to generate a heat map, which has a portion of it seen in the figure.

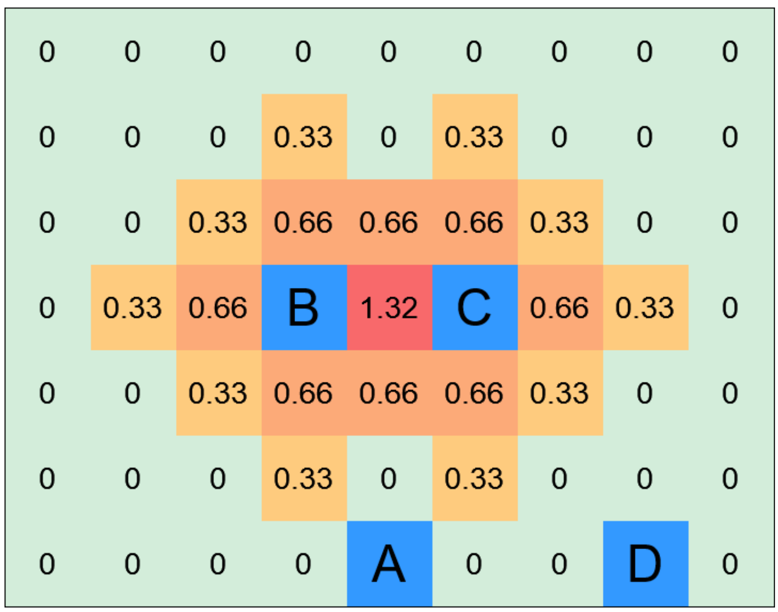


Figure 5: Heat Generated by Agents B and C for Agent A

The Heat Map Agent follows almost the same strategy as the Path-Aware strategy but takes the potential future conflicts into account; therefore, its cost estimation differs from the Path-Aware Agent. Unlike the Path-Aware strategy, which uses the actual cost of the paths, the Heat Map agent estimates the cost of the bids (i.e., paths) according to Algorithm 2. As seen, the overall cost is the sum of the estimated cost of each cell in the given path. That is, the cumulative cost is estimated by considering the relevant cell value on the heat map of each neighbor agent other than the current opponent agent. Line 2 iterates through the points in the given path whose cost is estimated, while Line 3 iterates through the heat maps of the neighbor agents except the opponent. The total cost is increased by each relevant cell value of the heat maps (Line 4). After calculating the estimated cost of each possible bid, the Heat Map Agent sorts them in descending order and employs the approach explained in Algorithm 1.

Algorithm 2: Bid Cost Estimation from Heat Map

Data:
H_T^j : Heat Map for Agent j at a given time T
N_c: Neighbor agents excluding the opponent in the field of view
P_c : A given path whose cost will be estimated
Output:
EC : Estimated cost of a given path, P_c

1 $EC \leftarrow 0$
2 **for** $i \leftarrow 1$ **to** $n \in |P_c|$ **do**
3   **foreach** $j \in N_c$ **do**
4         $EC \leftarrow EC + H_{t+i}^j.\text{get}(P_c^i)$
5   **end**
6 **end**
7 **return** EC

Throughout its life cycle in the environment, an agent can continuously observe its environment. By observing the changes in their environments, agents can deduce congestion in an area and behave in a fashion to avoid dense areas. One way to introduce congestion avoidance while designing the agent is to introduce a system to help the agent avoid bids that would direct it towards congested regions.

In this strategy, the agent utilizes the information available in the field of view to generate a simple heat map to perceive the crowd and avoid making bids that lead to those regions. The generated heat map will allow the agent to differentiate between offers by weighting bids that lead into crowded regions heavier than bids that do not.

CHAPTER IV

EVALUATION

In order to evaluate the proposed approach, we generated a variety of scenarios randomly and ran simulations on the given scenarios with different settings. We analyze the empirical results from different perspectives such as solution quality (our distributed solution versus centralized solutions), the impact of negotiation strategy, the effect of field of view (FoV) and the percentage of information shared with other agents. We conducted two different experimental studies and the corresponding results are discussed separately. Our initial experimental study is described in Section 4.1 and its extension is elaborately explained in Section 4.2.

4.1 Study 1

In our first study, we focus on the MAPF problem described as Case 1 in Chapter 3 and only evaluate the Path-Aware Agent for this type of problem. This section defines and discusses the findings on the first stage of our overall experiments.

4.1.1 Experimental Setup

To inspect the performance of decentralized approach against centralized approaches to resolve conflicts in MAPF, a comparison with two well-know centralized methods, namely Conflict-Based Search (CBS) [4] and CBS with the Weighted Pairwise Dependency Graph Heuristic and Rectangle Reasoning by Multi-Valued Decision Diagrams, named WDG+R in [29] is made. These centralized solutions are detailed below. Code of WDG+R is provided by it's authors. Each configuration contains sub-configurations depending on an agent's interaction with the environment. These are whether an agent has wait-action (no movement in a time step) and agent's behaviour when it reaches it's destination (leaving the environment). In decentralized approach varying Field of View sizes are also experimented in each configuration. Each scenario configuration experimented with 100 different scenarios in no obstacle 16x16 grid environments. The initial setup to test the performance design, we removed wait-action (no movement in a time step) from the action space of agents in all solvers to be suitable for the problem definition in Chapter 3. All experiments were carried on machines with the computing power of 16-Core 3.2 GHz Intel Xeon and 32 GB RAM. Each scenario configuration experimented with 100 different scenarios in no obstacle, 8x8, and 16x16 grid environments.

- **CBS:** Conflict-Based Search [4] is a two-level algorithm for centralized and optimal MAPF. At the low-level search, a single path is planned by an optimal shortest-path algorithm, like A*, under given constraints. A constraint is a tuple (i, v, t) where agent a_i is prohibited from occupying vertex v at time step t. At the high-level search, a constraint tree (CT) is operated to resolve conflicts between paths. CT is a binary tree of constraint nodes. Each CT node consists of a set of constraints for each agent. When a conflict is found between two agents, two child nodes are generated. In each child node, one agent in the conflict is prohibited from using conflicted vertex or edge by adding a constraint, and a new path is searched for that agent at the low level under the new constraint set.

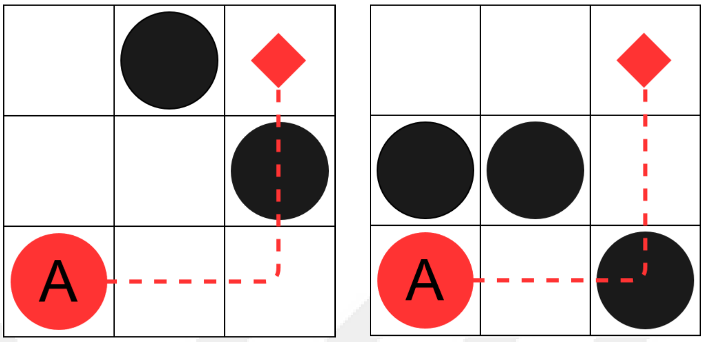
- **WDG+R:** Weighted Pairwise Dependency Graph Heuristic and Rectangle Reasoning by Multi-Valued Decision Diagrams is one of the recently enhanced variants of CBS and a state-of-art optimal MAPF solver. WDG+R operates smaller CTs by using an admissible heuristic in the high-level search named the Weighted Pairwise Dependency Graph (WDG) Heuristic. WDG represents the pairwise dependencies requiring some cost increase to resolve conflicts. Value of the minimum vertex cover of WDG serves as an admissible heuristic of a lower bound to cost increase to resolve conflicts. Besides, it efficiently resolves the rectangle conflicts by a reasoning technique introduced to CBS in [30]. A rectangle conflict occurs when two locations are required to be taken by both agents simultaneously, which means a certain cost increase to resolve the conflict. These enhancements provide a large factor of speedup compared to CBS.

Experiment scenarios are randomly generated from the configurations of MAPF benchmark datasets provided by [2] (i.e. grid size, number of agents, no obstacle cases). Table 1 provides the information of experimented MAPF scenarios. We set eight different problem configurations, which are 10, 15, 20, and 25-agent scenarios in an empty 8×8 grid, and 20, 40, 60, and 80-agent scenarios in an empty 16×16 grid. For each configuration, 100 different scenarios have experimented with randomly distributed path lengths between 2 and 14 for 8×8 grid scenarios, and 4 and 24 for 16×16 grid scenarios.

Table 1: Scenario Types

Configuration Name	**Grid Size**	**Number of Agents**	**Initial Path Range**
Config-1	8x8	10	2-14
Config-2	8x8	15	2-14
Config-3	8x8	20	2-14
Config-4	8x8	25	2-14
Config-5	16x16	20	4-24
Config-6	16x16	40	4-24
Config-7	16x16	60	4-24
Config-8	16x16	80	4-24

We set the runtime of CBS and WDG+R to 30 minutes 8×8 for grid scenarios and 1 hour for 16×16 grid scenarios in favor of obtaining optimal solutions for a rigorous evaluation of experiments, although CBS benchmarked in 5 minutes by [4] and WDG+R benchmarked in 1 minute by [29]. In our framework, the runtime metric does not represent the success capability of our decentralized solution since it would proceed in real-time. However, the run time of the negotiations in our systems is capped at 120 seconds. If a negotiation session fails to resolve within 120 seconds, that scenario is deemed to be unsolvable by the framework at that moment. Throughout our experiments, we have noted that a successful simulation instance takes 178.84 seconds on average, in 80 agent configurations. In addition, decentralized solution can fail to find a solution, when inactive agents close off movement to destination (Figure 6a), or surround others (Figure 6).



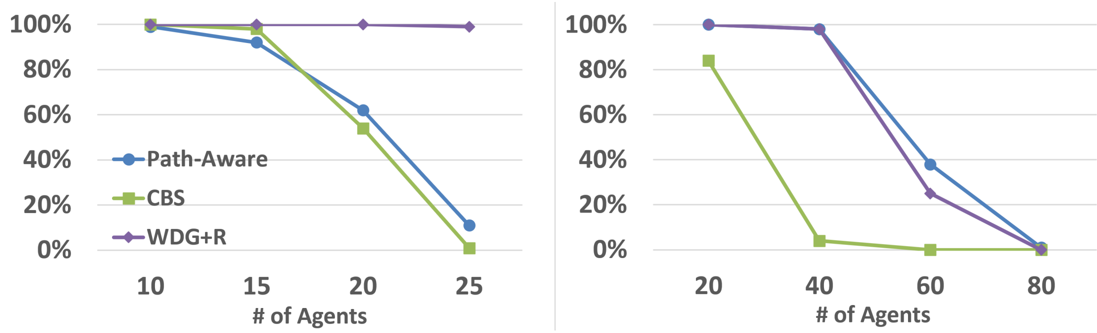
(a) (b)

Figure 6: Agent and Destination Blocked by Agents Reached Destinations

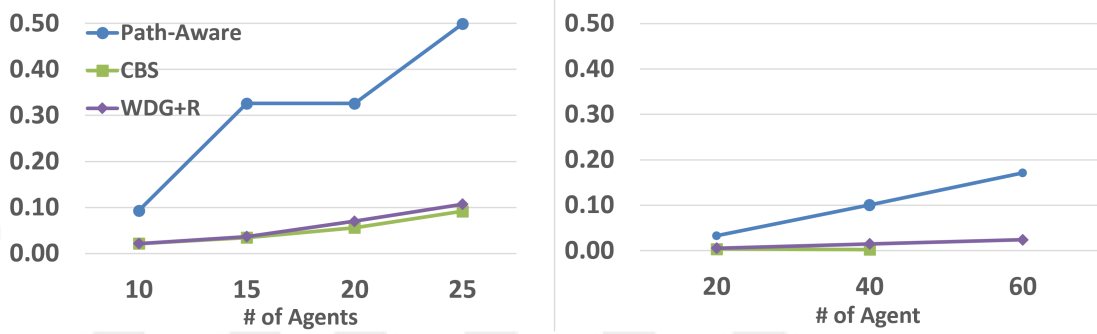
The field of view (FoV) is set to 2 for all agents to experiment with all 8×8 and 16×16 scenarios. Setting FoV to 1 corresponds that agents can be aware of conflicts just a one-time step before, limiting the practicality of negotiation to resolve conflicts. To observe the effect of FoV in our framework's solution performance, we repeat the experiments of Config-6 and Config-7 scenarios, setting FoV to 2, 3, and 4 for all agents. We do not test the effect of field of view in 8×8 grid environment since it is not much practicable to change the range, as it covers very large portion of the environment in bigger FoV values.

4.1.2 Experimental Results

Each metric for the results of each solution method is averaged over scenarios solved by itself throughout the evaluations in the following subsections. We determined the number of agents in the environment to such levels to observe remarkable breakdowns in success rates of CBS and WDG+R, which helps to see which MAPF problem complexity levels these centralized MAPF solution approaches become to fail at. 8×8 grid scenarios are experimented to benchmark CBS specifically and our proposed solution considering the performance evaluation of CBS



(a) Solution Rates of Scenarios, 8×8 on Left and 16×16 on Right



(b) Average Normalized Path Difference of Scenarios, 8×8 on Left and 16×16 on Right

Figure 7: Decentralized versus Centralized Approach Results

done in [4]. In 16×16 grid scenarios, we aim to see when the scaling capability of centralized and decentralized solutions are discriminated by changing the number of agents with large increments.

4.1.2.1 Decentralized versus Centralized Approach:

Solution rate (R) of the decentralized MAPF framework with Path-aware agents (PA) and the centralized solutions for 8×8 and 16×16 grids are represented in Figure 7a where the left chart corresponds for 8×8 grid results, and the right chart corresponds for 16×16 grid results. Although the decision complexity of agents is essential to measure the framework performance, this basic agent strategy outperforms CBS in 8×8 grid scenarios and also WDG+R in 16×16 grid scenarios in terms of R. However, PA results are not desirable 8×8 grid scenarios when the solution quality is considered according to the left chart in Figure 7b. To measure how much extra cost is produced to resolve conflicts in optimal paths, we use a metric named Average Normalized Path Difference (D_{avg}), which is equal to $(C_1$-$C_0)/k$ where C_0 is the cost of initial paths of all agents and C_1 is the sum of

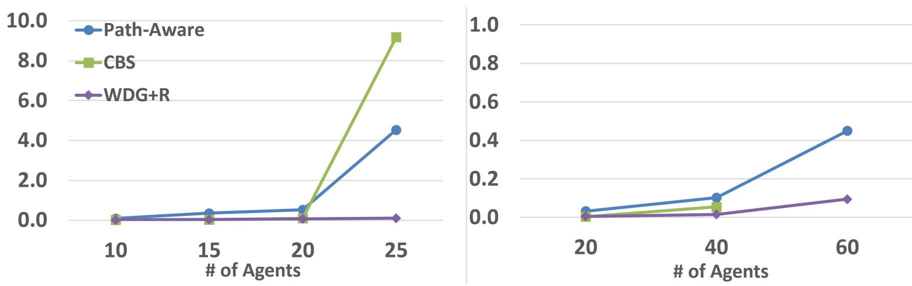


Figure 8: Average Normalized Path Difference / Solution Rate

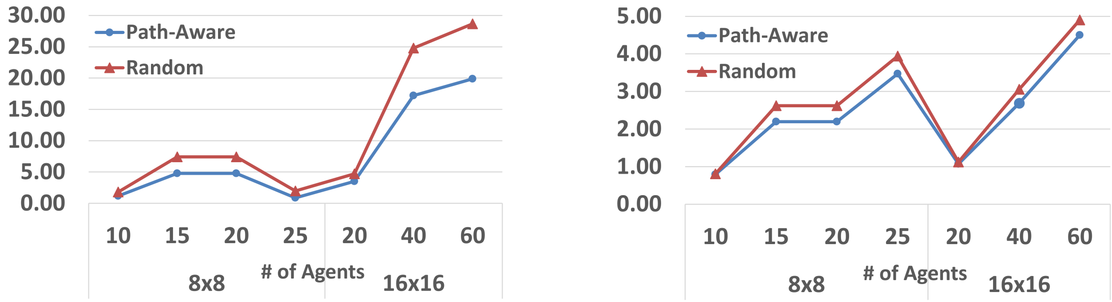
individual path costs (SIC) value attained in a solution. This metric means how much-added cost is yielded to resolve conflicts compared to C_0. D_{avg} gives the information of how much cost increase occurs in which environments for the side of the self-interested agent only consider its own cost valuation based on its initial path cost. We use D_{avg} in our evaluation instead of Optimality Gap due to CBS not yielding optimal solutions in most of high density and large environment configurations. 7b shows that PA performs well in the scenarios of the high number of agents in larger maps, which indicates the scalability of the decentralized solution compared to centralized solutions. However, only one scenario of Config-8 was solved by PA because agents cannot negotiate to resolve a conflict caused due to the demonstrated situations in Figure 6. We note that Config-8 scenarios are not evaluated in the following metrics since optimal solutions could not be obtained for them.

As all scenarios are not solved by WDG+R, we do not have the complete information to measure the solution quality of the decentralized solutions since optimal solutions are taken as the basis for it. It is not a health assessment to compare the cases that CBS and WDG+R can solve with the scenarios they cannot solve, but PA solves. For this reason, there is a need for a variable that shows the relationship between solution rates and normalized path differences more dynamically, which is D_{avg}/R. Figure 8 has enabled dynamic changes to be observed in a wide range with D_{avg}/R. Although CBS seems to be more successful than PA in 8×8 grid scenarios in terms of D_{avg}, Figure 8 shows that Path-Aware agents are more successful when considering the performances in new metric solution rates. It is observed that CBS fails dramatically in scenarios involving 25 Agents. When the 8×8 and 16×16 grid scenarios are compared, it is observed that the performance of the WDG+R remained the same, while the performance of the Path Aware Agents was 10 times better. This result shows that in larger domains, decentralized and intelligent agents such as Path-Aware have less performance difference from optimal solvers and high privacy support.

4.1.2.2 Effect of Intelligence of Agents:

To figure out the effect of intelligence of agents in our framework, a baseline representing a random decision behavior for the negotiation protocol is needed. Therefore, we present a basic decision mechanism adapted by an experimental agent named *Random Agent*. Random Agent accepts its opponent's offer with %50 probability. Otherwise, it repeats its previous offer with %50 probability if it has enough tokens. It generates its offer space exactly in the same way as the Path-aware strategy. The main difference is about accepting and deciding the usage of tokens. We highlight that Random Agent provides a lower bound performance for any prospective self-interested rational agent designed for the proposed decentralized MAPF framework.

Figure 9a shows the total number of token exchanges in a session for both agent types. Negotiation between Path-Aware agents results in less number of token exchanges compared to negotiations of Random agents, which shows that Path-Aware agents insist on their offers only to maintain their own cost balance, whereas Random agents insist or concede randomly. Negotiations between Path-Aware agents reach an agreement faster than random agents in all environments, as seen in Figure 9b. The number of negotiations by Random agents represents a baseline for the negotiation protocol if the agent decides indifferent to counter's bids. So, it can be concluded that when agents behave more analytical, they can reach an agreement faster with our proposed negotiation protocol. Since the solution rates of the decentralized solution with Random agents (RA) and the decentralized solution with Path-Aware agents (PA) is low, we do not seek a



(a) Total Number of Token Exchange (b) Number of Negotiations/Agent

Figure 9: Path-Aware Agent versus Random Agent

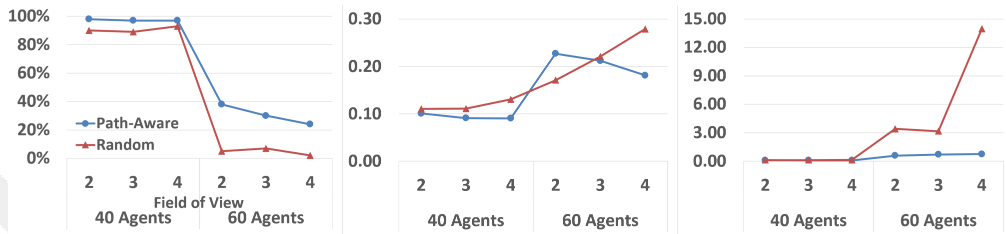


Figure 10: (a) Solution Rate with FoV, (b) Average Normalized Path Difference with FoV, (c) Average Normalized Path Difference / Solution Rate with FoV

trend for the curves in Figure 9.

4.1.3 Effect of Field of View:

We experiment with 40 and 60-agent scenarios in 16×16 grid with different FoV values to figure out how the perception and information broadcast range of agents relate to the solution performance of the decentralized MAPF approach. This relation branches in three aspects, R, D_{avg}, and D_{avg}/R. Figure 10 presents the related curves for PA and RA. Change in FoV of Random agent has no general trend in R in all scenarios, while the increase in FoV decreases R in all scenarios when all agents are Path-Aware. This decrease is 1% for 40-agent scenarios and 14% for 60-agent scenarios, which shows that Path-Aware agents struggle to agree in dense environments when they have a wider FoV. On the other hand, Path-Aware agents can have better paths with wider FoV according to D_{avg} trend in Chart B. Besides, the third chart in Figure 10 shows that the solutions achieved with PA are much better in terms of D_{avg}/R compared to RA solutions. So, it can be concluded that intelligent agents can preserve their own path cost interest by negotiating with others under TAOP. This output is expected because Path-Aware agents change their bidding behavior based on current cost analysis (of remaining path length and the remaining token amount at the negotiation time). This evaluation becomes more useful if more information about the environment is used.

In large FoV cases, Random agents tend to perform worse as their decision-making process is stochastic. As the accepted offers are registered, having a large FoV thus results in more constraints for agents, which reduces path search space. Path-Aware Agents provide solutions to these problems both in acceptance and by spending their tokens correctly. On the other hand, Random Agents do not have a specific strategy other than entering into a negotiation and accepting long paths randomly in the face of these problems. This situation is reflected in the average number of negotiations per agent. When looking at the difference of the average negotiation per agent between FoV 2 and FoV 3, Random agents (2.17) are 2.59 times more than Path-Aware agents (0.83).When the same variable is examined between FoV 3 and Fov 4, Random agents (2.11) increased 5.90 times more than Path-Aware agents (0.35).

4.2 Study 2

This section defines and discusses the findings on the second stage of our experiments, where we follow up and advance on our findings in Section 4.1. In other words, this study covers the extended analysis of four variants of MAPF problem that are defined in Chapter 3, as well as comparison of the proposed negotiation strategies, "Path-Aware" and "HeatMap". Accordingly, Section 4.2.1 describes the experimental set up while Section 4.2.2 reports and discusses the results.

4.2.1 Experimental Setup

Following the results obtained in Section 4.1, we have expanded our experiment setup to include wait action and agent disappearing the environment when it reaches its goal, which we had left out in our previous study. Wait action allows agents to stay at a location in a time step. Leave on target behavior makes it such that when agents reach their destinations, they do not become obstacles for other agents. Based on our previous experimental results, we observed that 16×16 scenarios are more challenging for the agents and make it possible to observe the effects of Field of View on our experiments. Therefore, we only focus on such scenarios for this study. Each scenario is run five times due to the stochastic behaviour of the solution approach, and every scenario is consisting of same agent type. To achieve compatibility between studies, we have used the same environment setups used in Section 4.1, and made each agent start with 10 tokens in the beginning of scenarios.

Since we report that CBS usually fails to find out optimal solutions in these scenarios, we use the Explicit Estimation Conflict-based Search (EECBS) [31] as an optimal solver for benchmarks. EECBS is the state-of-art variant of the Conflict-based Search algorithm for optimal MAPF. It is a bounded sub-optimal planning algorithm that guarantees a solution cost by an optimality factor of w. It scales to larger instances; however, its scalability is limited by the number of agents influencing the solution time exponentially. It uses a focal search instead of the best first search in A* at a low level, where solutions have a cost value of at most $w \cdot f$ where f is the lower bound to the optimal solution cost found during the search. This system uses Explicit Estimation Search (ESS) [32] on a high-level search to reach a suboptimal solution faster by utilizing inadmissible information about potential cost increments, and it is more likely to improve the lower bound on the solution cost compared to the previous bounded-suboptimal CBS, Enhanced CBS [33], according to the benchmark in [31]. The EECBS used in the experiments includes several enhancements to CBS and is referred to as EECBS+ in [31].

4.2.2 Experimental Results

We evaluate the performance of the proposed decentralized approaches in each case by comparing their performance with those of the EECBS (i.e., centralized approach) with respect to solution rate. Solution rate shows the agent's capability of finding a solution in given configurations on the scenarios. It corresponds to the rate of the runs successfully solved by the given approach.

Furthermore, we compare the performance of the proposed negotiation strategies in the decentralized negotiation solutions with respect to the optimality gap, information sharing rate, average number of negotiations, and number of tokens exchanged. For all metrics mentioned here, we only consider the results of the runs where both strategies found a solution to evaluate the strategies fairly. Table 2 shows the number of cases solved by both Path-Aware and Heat Map Agents for each case that the EECBS has solved. As it can be seen obviously, both agent strategies solve all the scenarios successfully when the number of the agents is 20. As the number of the agents increases in the 16×16 grid environment, the number of successful runs dramatically drops. It is more challenging to find a solution for the stay at target MAPF scenarios (Case 1 and 2) since the agents become obstacles for others when they reach their destinations.

	Number of Agents			
	20	**40**	**60**	**80**
Case 1	100	91	8	0
Case 2	100	91	4	0
Case 3	100	99	66	19
Case 4	100	100	62	16

Table 2: Number of cases solved by Path-Aware Agent (FoV2), Heat Map Agent (FoV2) and EECBS

The optimality gap describes the rate of difference between the average path cost of all agents provided by the chosen approach and that of the optimal solution provided by EECBS as shown in Equation 1 where $|\pi^*_{solution}|$ is the average path length of the agents provided by EECBS.

$$OptimalityGap = \frac{|\pi_{solution}| - |\pi^*_{solution}|}{|\pi^*_{solution}|} \tag{1}$$

We introduce a metric to measure to what extent the privacy of the agents is concerned, which we called *information share rate.* This metric considers the extent of an overall path shared with other agents within the given scenario. As shown in Equation 2, we calculate the rate of the shared path with other agents per each agent and take the average of them where $\pi^i_{solution}$ and $\pi^i_{broadcasted}$ denote the final path of the Agent i and the part of that path shared with other agents during the broadcast, respectively. Recall that we have k agents in total.

$$InformationShareRate = \frac{1}{k} * \sum_{i=1}^{k} \frac{|\pi^i_{solution} \cap \pi^i_{broadcasted}|}{|\pi^i_{solution}|} \tag{2}$$

In the following sections, we first investigate the effect of the wait action and disappear at target on the performance of the proposed decentralized negotiation approaches (Section 4.2.2.1), and then compare the performance of the proposed approach with the state-of-the-art centralized approach (Section 4.2.2.2).

4.2.2.1 Effect of Wait Action and Disappear at Target in Decentralized

Experiments in Section 4.1 showed that Path-Aware agent is capable of providing solutions in challenging settings where agents have to take action in each time step and act as an obstacle for other agents in the system when they reach their destinations. It would be interesting to see how the proposed negotiation approach handles the same scenarios if we enable agents to wait at any time and or disappear from the environment when they reach their goals. Consequently, we evaluate the performance of the proposed approaches on the previously generated 16×16 scenarios for four different variants of the MAPF problems as described in Chapter 3.

Figure 11 depicts the scenario solution rates of both agents on different environment densities (i.e., different number of agents in the grids: 20, 40, 60, 80) and with different field of view sizes (e.g., 2, 3, 4). The positive effect of enabling wait action and disappear at target on the solution rate of agents can be observed in Figure 11. Since we observe that agents have low solution rates when the field of view size is 4, we have omitted results of this configuration in the following evaluations for clarity.

Both agents perform much better when disappearing at target, especially for the scenarios consisting of 40, 60, and 80 agents. That makes sense since the complexity of the problem is reduced since the agents at targets are not obstacles for others anymore. Another observation is that the performance of the Path-Aware agent is affected more positively when the agents have wait action. Unlike the Path-Aware agent, the decision of the Heat Map agent is influenced by its surroundings more during the bidding phase. Therefore, the Heat Map agent may prefer to take more wait action, which may lead that agent to get stuck.

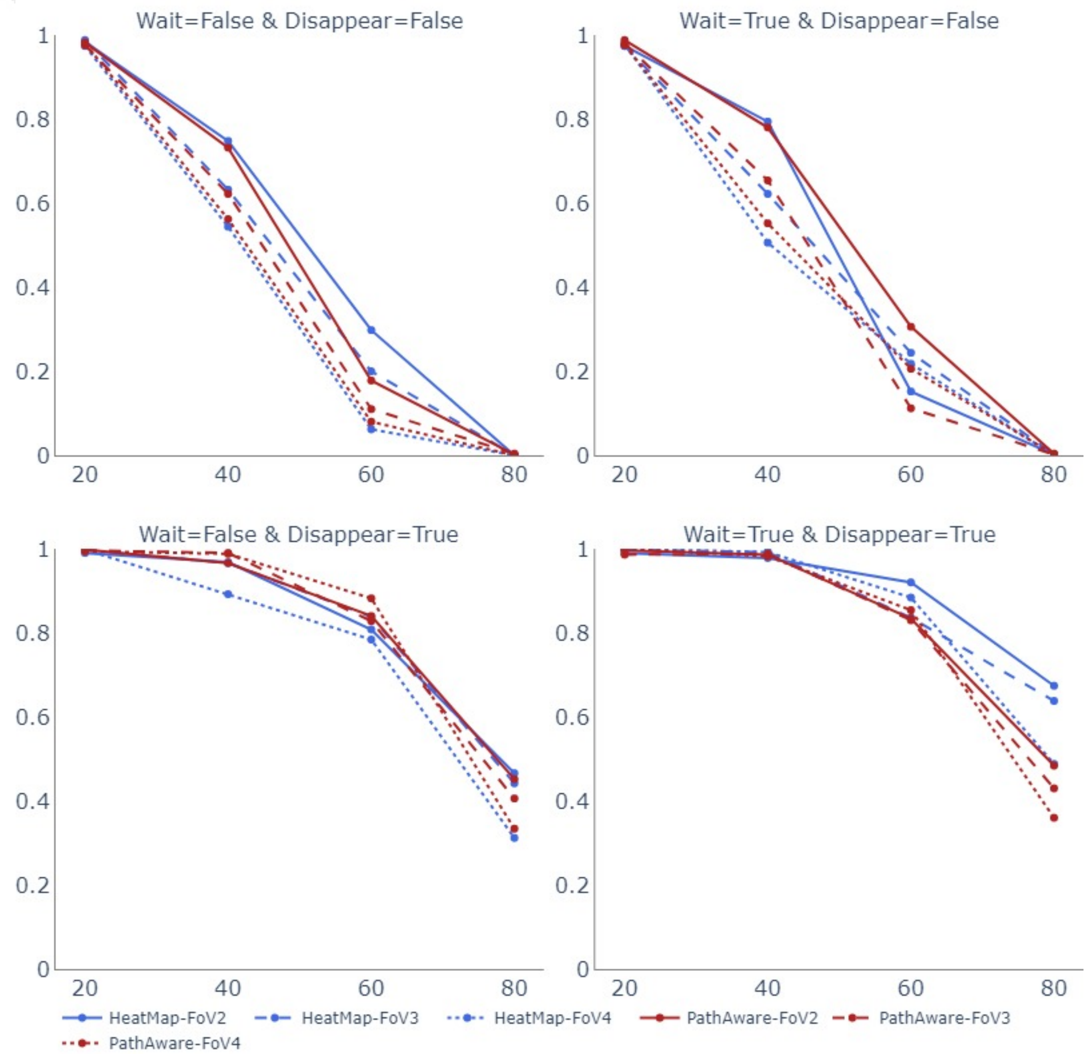


Figure 11: Solution Rate Comparison Between Heat Map and Path-Aware Agents

Figure 12 shows the average optimality gap for both agents in four cases with different field of view sizes. It should be noted that the optimality gap comparison can only be made on scenarios where the agents can find a solution with the given configuration. We can observe that allowing wait action has a seemingly negative impact on the optimality gap of the negotiation approaches, although it may not be valid for all cases. However, this trade-off between the success rate and the optimality of the solutions allows agents to resolve more scenarios. In the disappear at target case, agents even can resolve the most challenging scenarios consisting of 80 agents, whereas they provide more optimal solutions on the scenarios consisting of 20, 40, 60 agents. Furthermore, it can be seen that the Heat Map agent can resolve cases with a smaller optimality gap than the Path-Aware agent overall.

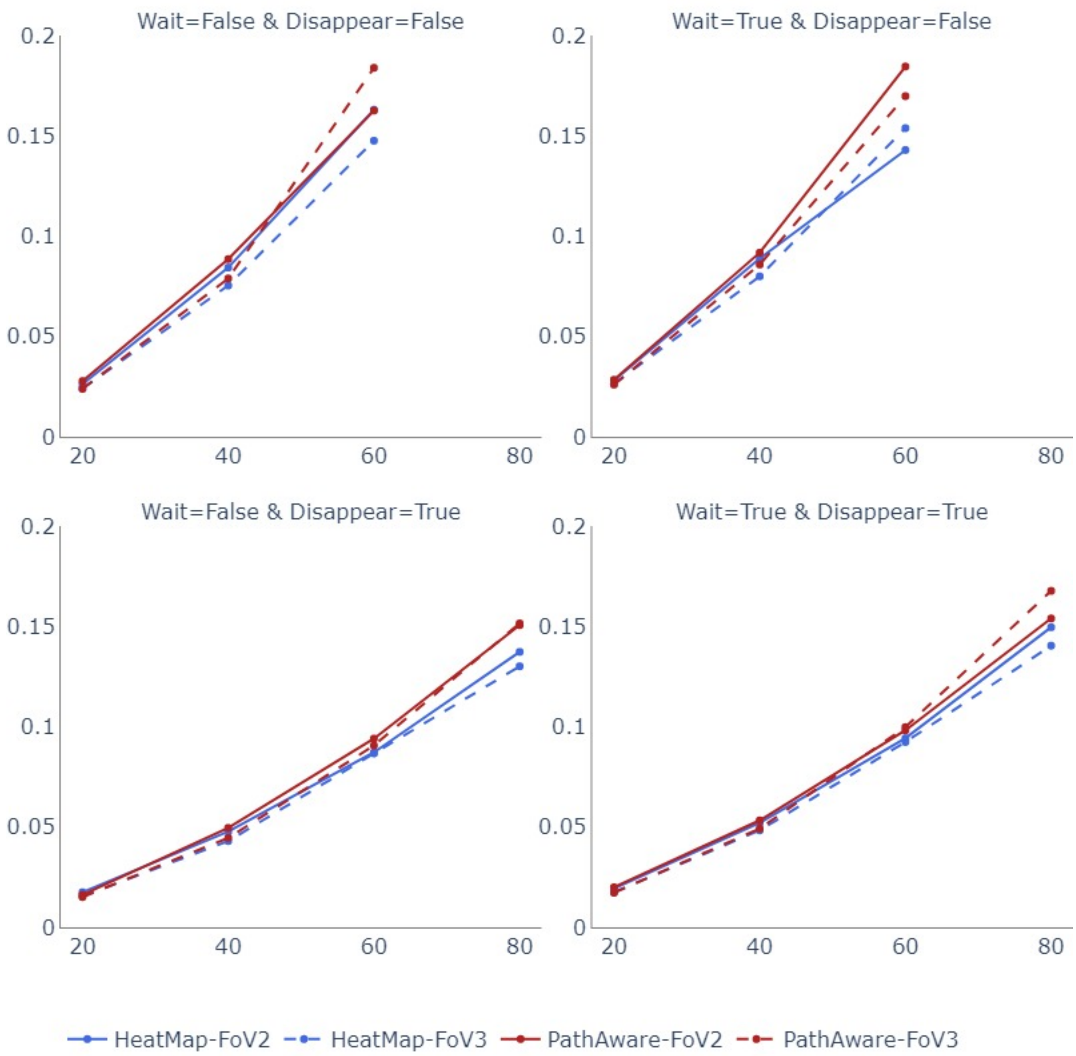


Figure 12: Optimality Gap Comparison of Heat Map and Path-Aware Agents

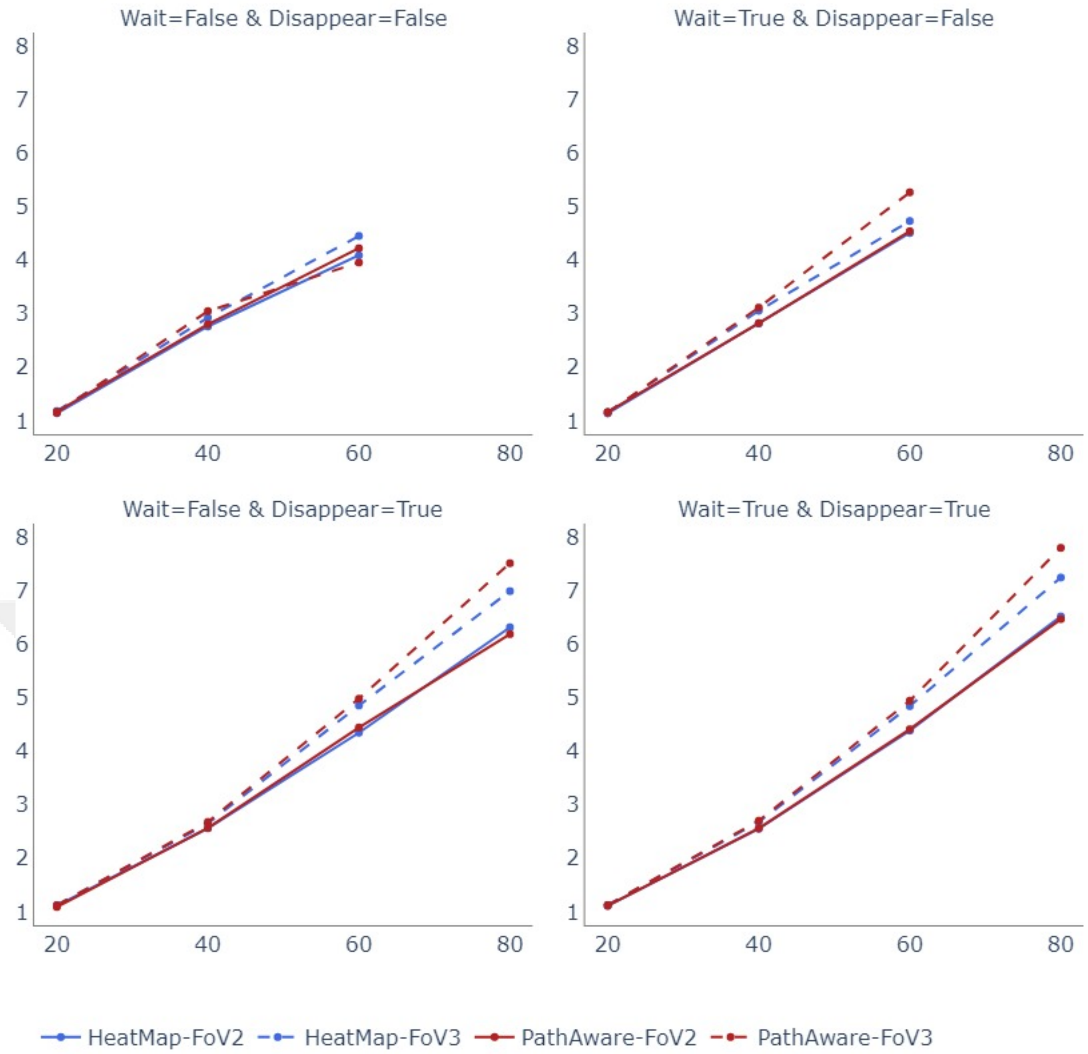


Figure 13: Average Number of Negotiations per Agent

Figure 13 and Figure 14 show the average number of negotiations per agent and average number of tokens exchanged in those negotiations, respectively. When the field of view is two, almost the same number of negotiations per agent took place for both agents. On the other hand, when the field of view is three, it seems that the Heat Map agent engages in slightly fewer negotiations overall compared to the Path-Aware agent. This supports that the results of the Heat Map agent's negotiations lead to fewer conflicts throughout the simulation. Recall that the Heat Map agent avoids potential future conflicts and may tend to choose a longer path. We can further support this argument by looking back at Figure 11 to recall that the Heat Map agent is able to resolve a considerable amount of cases against

Path-Aware. Since the usage of the tokens is almost the same, there is no significant difference between the average tokens exchanged by both agents.The slight difference in the average number of exchanged tokens may stem from the average number of negotiations that agents encountered.

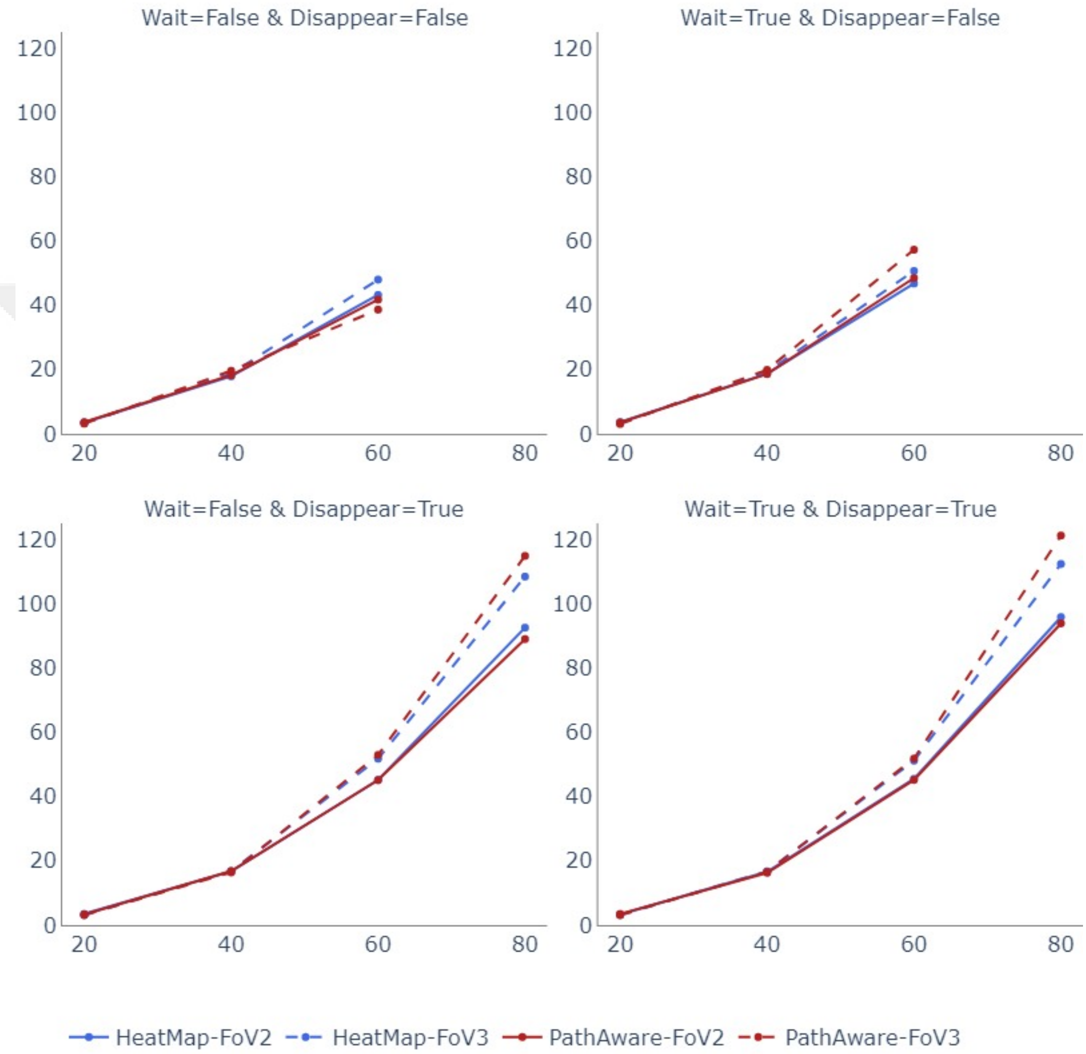


Figure 14: Total Number of Tokens Exchanged

For resolving the conflicts they encounter on their own in decentralized systems, they need to reveal some portion of their path information to other agents. Figure 15 shows the average information share rate for the Path-Aware and the Heat Map Agents. We might expect to see a decrease in the information share rate when dealing with a MAPF problem where the agents disappear at their targets because they are not obstacles for others. However, once their targets become free when they disappear from the environment, other agents could utilize the cells which became free because of their disappearance. Utilizing the freed cells may lead them to interact with other agents more, causing increased information sharing. Consequently, it may increase the information share rate. It can be seen that the average information rate has a slight increase when the agents disappear when they reach their destinations. Moreover, we can observe that the average information share rate is higher when the field of view is larger irrespective of the condition as is expected, since a larger field of views make agents perceive more neighbor agents.

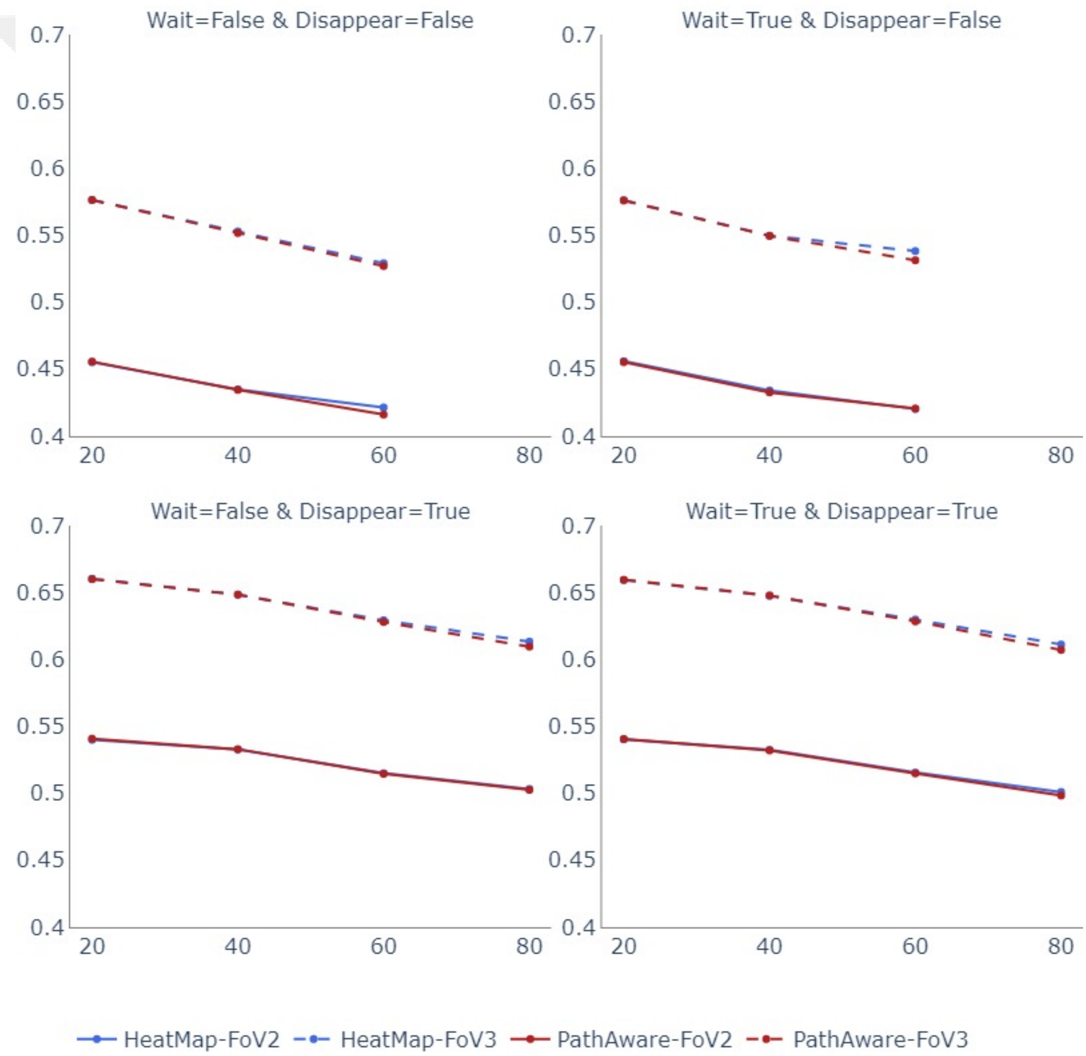


Figure 15: Average Information Share Rate per Agent

4.2.2.2 Decentralized versus Centralized Approach

To compare the performance of our distributed negotiation approaches with the centralized solutions, we have used EECBS to find optimal solutions on all MAPF configurations. We found that agents have a higher solution rate on average when the field of view is 2. Thus, we consider those results while comparing our approaches with the centralized ones.

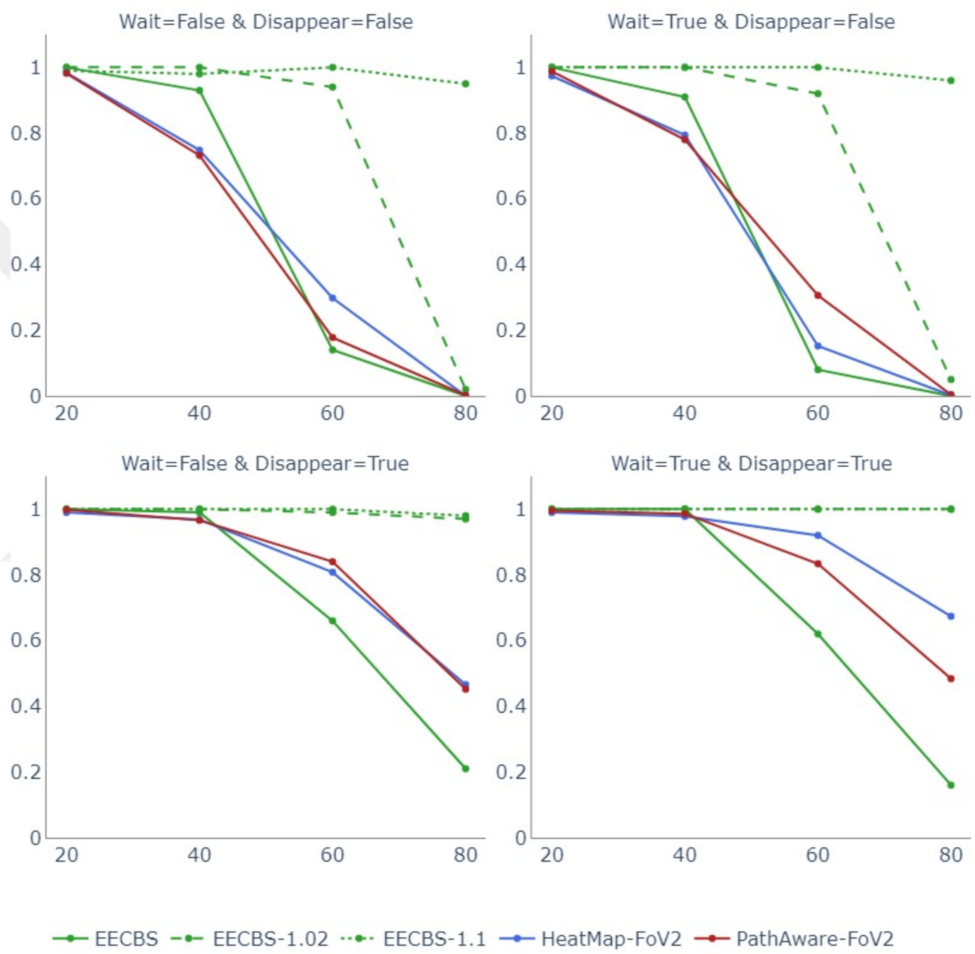


Figure 16: Solution Rate for EECBS on Different Suboptimalities and Agents on FoV2

Figures 16 and 17 display the solution rate and optimality gap values respectively for the EECBS with suboptimal variants, Path-Aware agent and Heat Map agent on field of view size of 2. Similar to the results for Case 1 (i.e., no wait action and stay at target scenarios) in Section 4.1, the decentralized approaches seem to have a higher solution rate compared to the optimal centralized solution approach

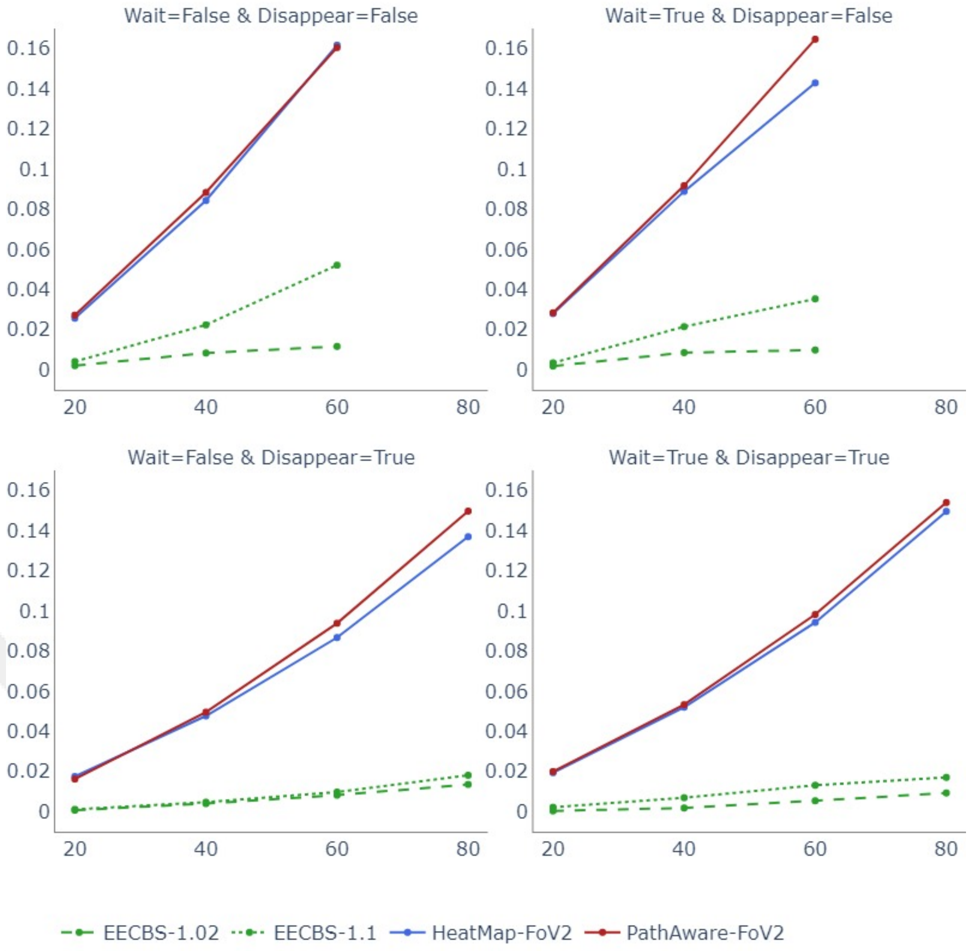


Figure 17: Optimality Gap of EECBS on Different Suboptimalities and Agents on FoV2

(i.e., EECBS), especially in scenarios with more agents. Recall that EECBS may not find optimal solutions for some scenarios. On the other hand, when EECBS is allowed to perform with a degree of suboptimality, it can provide solutions for the majority of the scenarios, and its suboptimal variants have a higher solution rate than our approach. While the decentralized agents can provide more solutions in high-density scenarios (e.g., when the number of agents is 60 or 80) compared to optimal EECBS, the optimality gap in which these scenarios are resolved are at most 16% less optimal. Moreover, we can see that extending Path-Aware agent and developing the Heat Map agent has improved the overall solution rate and slightly decreased the optimality gap on decentralized setup. Since the Heat Map agent is designed to be more aware of the environment, these results showed that a more sophisticated agent design might yield better performance.

Lastly, we investigate the trade-off between privacy and success rate for the proposed negotiation approaches. Considering this trade-off, we introduce a privacy-sensitive success criterion denoted by μ. Equation 3 shows how we calculate this metric where α denotes the weight of the privacy while (1-α) indicates the weight of the solution rate. When the agents are more sensitive about their privacy, higher values of α are chosen to estimate the trade-off, and when they care about the solution rate, the other way around. Accordingly, we pick three different α values, 0.5, 0.7, and 0.9 to weigh the sensitivity of privacy.

$$\mu = ((1-\alpha) * SolutionRate) + (\alpha * (1 - InformationShareRate)) \tag{3}$$

Figures 18 shows the performance of the agents with respect to the privacy-sensitive success criterion defined above, μ. When the agents can take wait action and disappeared from the target, it seems that the Path-Aware agent performs better than the Heat Map Agent. On the other hands, when the agents can still have the wait action but stay at their target, the performance of the Heat Map Agent seems higher than that of the Path-Aware agent especially for the high sensitivity to the privacy.

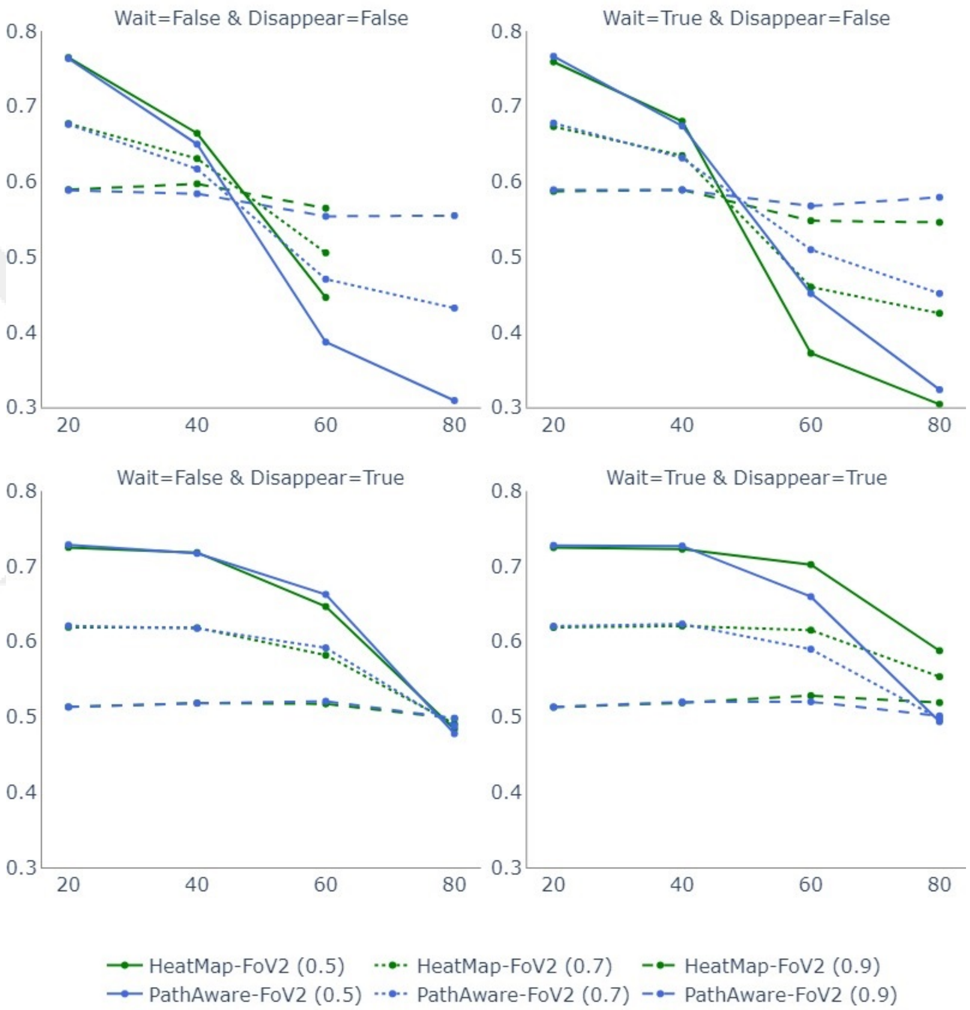


Figure 18: Trade-off Between Success Rate and Info Sharing

CHAPTER V

CONCLUSION

This thesis addresses how self-interested agents can coordinate in a grid environment to reach their destination without any collision and proposes solving the conflicts on the paths by means of bilateral negotiations. Accordingly, we propose a novel negotiation protocol and two compatible negotiation strategies: Path-Aware and Heat Map. While the Path-Aware strategy has more concern about the length of its final path, the Heat Map strategy considers the potential future conflict among agents and exploits the broadcasted path information by other agents accordingly.

According to the initial experimental results for the case where the agents do not have wait action and stay at their target when they reach their destinations, the proposed approach with Path-Aware strategy enables agents to optimize their paths in real-time without sharing their complete path information with everyone. This problem is harder to solve, especially when the grid size gets more extensive and dense (i.e., many agents and long paths per agent). In such cases, the proposed approach has the edge over optimal centralized solution approaches in terms of solution rate. However, when the optimality of the decentralized solutions are compared with sub-optimal solutions of centralized solvers, the need for improvement is evident. The experimental evaluation analysis showed that the Path-Aware negotiation approach finds reasonably good solutions in most cases and performed better on the aforementioned challenging scenarios than the centralized solution such as CBS and WDG+R for optimal solutions search.

According to the second experimental results where we investigate different variants of the multi-agent path finding problem, overall, the Heat Map agent performs better than the Path Aware negotiation approach and a centralized approach like EECBS aiming to find only the optimal solutions when agents have wait actions and disappear at their targets. However, the EECBS with suboptimal variants outperformed the proposed approaches with both solution and optimality rates. On the other hand, we emphasize that our approach concerns the agents' privacy; therefore, there is a trade-off between privacy and performance, which could be investigated further. Similar to the previous results, negotiation-based approaches find good solutions, but there is room for improvement as far as the optimality of the solution is concerned. Depending on the variants of the MAPF, the Path-Aware and Heat Map Agents perform well in terms of a success criterion taking both solution rate and privacy of the agents into account.

As future work, we are planning to extend our approach by adopting multilateral negotiation instead of multiple consecutive bilateral negotiations and to compare its performance with the proposed approach. In the current work, agents move in time steps however should also be able to move constantly in line with their path. Furthermore, designing more sophisticated agents that are capable of thinking ahead in their negotiation and path planning processes to improve optimality and solution rate of the system is necessary. Additionally, allowing the system to continue despite negotiation failures can be considered. Without a doubt, the trade-off between privacy and performance should be further investigated more elaborately.

Bibliography

[1] R. Stern, N. Sturtevant, A. Felner, S. Koenig, H. Ma, T. T. Walker, J. Li, D. Atzmon, L. Cohen, T. K. S. Kumar, E. Boyarski, and R. Barták, "Multi-agent pathfinding: Definitions, variants, and benchmarks," *CoRR*, vol. abs/1906.08291, 2019.

[2] R. Stern, *Multi-Agent Path Finding – An Overview*, pp. 96–115. Cham: Springer International Publishing, 2019.

[3] M. Erdmann and T. Lozano-Perez, "On multiple moving objects," in *Proceedings. 1986 IEEE International Conference on Robotics and Automation*, vol. 3, pp. 1419–1424, 1986.

[4] G. Sharon, R. Stern, A. Felner, and N. R. Sturtevant, "Conflict-based search for optimal multi-agent pathfinding," *Artif. Intell.*, vol. 219, pp. 40–66, 2012.

[5] V. R. Desaraju and J. P. How, "Decentralized path planning for multi-agent teams with complex constraints," *Auton. Robots*, vol. 32, p. 385–403, May 2012.

[6] G. Weiss, *Multiagent Systems*. The MIT Press, 2013.

[7] E. Hoz, J. M. Giménez-Guzmán, I. Marsá-Maestre, and D. Orden, "Automated negotiation for resource assignment in wireless surveillance sensor networks," *Sensors (Basel, Switzerland)*, vol. 15, pp. 29547 – 29568, 2015.

[8] I. Marsá-Maestre, M. Klein, C. Jonker, and R. Aydogan, "From problems to protocols: Towards a negotiation handbook," *Decis. Support Syst.*, vol. 60, pp. 39–54, 2014.

[9] D. Jonge, F. Bistaffa, and J. Levy, "A heuristic algorithm for multi-agent vehicle routing with automated negotiation," in *20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021)*, 05 2021.

[10] R. Aydoğan, T. Baarslag, K. Fujita, J. Mell, J. Gratch, D. de Jonge, Y. Mohammad, S. Nakadai, S. Morinaga, H. Osawa, C. Aranha, and C. M. Jonker, "Challenges and main results of the automated negotiating agents competition (ANAC) 2019," in *Multi-Agent Systems and Agreement Technologies* (N. Bassiliades, G. Chalkiadakis, and D. de Jonge, eds.), (Cham), pp. 366–381, Springer International Publishing, 2020.

[11] H. Inotsume, A. Aggarwal, R. Higa, and S. Nakadai, "Path negotiation for self-interested multirobot vehicles in shared space," in *Proceedings of the International Conference on Intelligent Robots and Systems*, (Las Vegas, NV, USA), pp. 11587–11594, IEEE, 2020.

[12] A. Pritchett and A. Genton, "Negotiated decentralized aircraft conflict resolution," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, pp. 81–91, 2018.

[13] O. Purwin, R. D'Andrea, and J. Lee, "Theory and implementation of path planning by negotiation for decentralized agents," *Robotics Auton. Syst.*, vol. 56, pp. 422–436, 2008.

[14] A. Felner, R. Stern, S. E. Shimony, E. Boyarski, M. Goldenberg, G. Sharon, N. R. Sturtevant, G. Wagner, and P. Surynek, "Search-based optimal solvers for the multi-agent pathfinding problem: Summary and challenges," in *SOCS*, 2017.

[15] O. Amir, G. Sharon, and R. Stern, "Multi-agent pathfinding as a combinatorial auction," in *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, p. 2003–2009, AAAI Press, 2015.

[16] A. Gautier, B. Lacerda, N. Hawes, and M. Wooldridge, "Negotiated path planning for non-cooperative multi-robot systems," in *IJCAI*, 2020.

[17] P. B. Sujit, A. Sinha, and D. Ghose, "Multiple uav task allocation using negotiation," in *AAMAS '06: Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, AAMAS '06, (New York, NY, USA), p. 471–478, Association for Computing Machinery, 2006.

[18] D. C. Parkes, "Ibundle: An efficient ascending price bundle auction," in *Proceedings of the 1st ACM Conference on Electronic Commerce*, EC '99, (New York, NY, USA), p. 148–157, Association for Computing Machinery, 1999.

[19] T. Roughgarden and É. Tardos, "How bad is selfish routing?," *J. ACM*, vol. 49, pp. 236–259, 2002.

[20] R. J. Cole, Y. Dodis, and T. Roughgarden, "Pricing network edges for heterogeneous selfish users," in *STOC '03*, 2003.

[21] Z. Bnaya, R. Stern, A. Felner, R. Zivan, and S. Okamoto, "Multi-agent path finding for self interested agents," in *Sixth Annual Symposium on Combinatorial Search*, 2013.

[22] R. I. Brafman and C. Domshlak, "From one to many: Planning for loosely coupled multi-agent systems," in *ICAPS*, 2008.

[23] A. Bonisoli, A. Gerevini, A. Saetti, and I. Serina, "A privacy-preserving model for the multi-agent propositional planning problem," in *ECAI*, 2014.

[24] R. Aydoğan, K. V. Hindriks, and C. M. Jonker, *Multilateral Mediated Negotiation Protocols with Feedback*, pp. 43–59. Tokyo: Springer Japan, 2014.

[25] R. Aydoğan, D. Festen, K. Hindriks, and C. Jonker, "Alternating offers protocols for multilateral negotiation," *Studies in Computational Intelligence*, vol. 674, pp. 153–167, Jan. 2017.

[26] J. S. Rosenschein and G. Zlotkin, *Rules of Encounter: Designing Conventions for Automated Negotiation among Computers.* Cambridge, MA, USA: MIT Press, 1994.

[27] M. Klein, P. Faratin, H. Sayama, and Y. Bar-Yam, "Protocols for negotiating complex contracts," *IEEE Intelligent Systems*, vol. 18, pp. 32–38, 12 2003.

[28] T. Baarslag, E. H. Gerding, R. Aydoğan, and M. Schraefel, "Optimal negotiation decision functions in time-sensitive domains," in *2015 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*, vol. 2, pp. 190–197, Dec 2015.

[29] J. Li, A. Felner, E. Boyarski, H. Ma, and S. Koenig, "Improved heuristics for multi-agent path finding with conflict-based search," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pp. 442–449, International Joint Conferences on Artificial Intelligence Organization, 7 2019.

[30] J. Li, D. Harabor, P. J. Stuckey, H. Ma, and S. Koenig, "Symmetry-breaking constraints for grid-based multi-agent path finding," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 6087–6095, Jul. 2019.

[31] J. Li, W. Ruml, and S. Koenig, "Eecbs: A bounded-suboptimal search for multi-agent path finding," in *AAAI*, 2021.

[32] J. Thayer and W. Ruml, "Bounded suboptimal search: A direct approach using inadmissible estimates," in *IJCAI*, 2011.

[33] M. Barer, G. Sharon, R. Stern, and A. Felner, "Suboptimal variants of the conflict-based search algorithm for the multi-agent pathfinding problem," in *SOCS*, 2014.

VITA

Cihan Eran received the B.Sc. degree in Computer Science from Özyeğin University, Istanbul, Turkey, in 2018. He started his M.Sc. degree as a research assistant under the supervision of Professor Reyhan Aydoğan in the Computer Science department at Özyeğin University, Istanbul, Turkey. His main research interests include multi-agent systems and multi-agent path planning.

His conference paper titled "A Decentralized Tokenbased Negotiation Approach for Multi-Agent Path Finding " co-written by Onur Keskin, Furkan Canturk and Reyhan Aydoğan has been selected as runner-up for the 18th European Conference on Multi-agent Systems (EUMAS-2021) Best Paper Award.